z/OS
Version 2 Release 4

Cryptographic Services
PKI Services Guide and Reference

IBM

SA23-2286-40

Note

Before using this information and the product it supports, read the information in "Notices" on page 705.

This edition applies to Version 2 Release 4 of z/OS (5650-ZOS) and to all subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2020-08-24

Contents

Figures

Tables

About this document

This document supports z/OS® (5650-ZOS). This document contains information about planning, customizing, administering, and using the PKI Services component of the z/OS Cryptographic Services.

PKI Services provides a certificate authority for the z/OS environment and enables you to issue and administer digital certificates, so that you do not have to purchase them from an external certificate authority. This document provides you with the information you need to become productive with PKI Services. It contains information about the following topics:

- Procedures for setting up PKI Services on the z/OS platform.
- Using the PKI Services administration and user web pages, you can easily issue digital certificates to trusted parties and control whether a certificate is renewed or revoked.
- Guidelines to help you plan for PKI Services, such as how to integrate PKI Services components with other products installed at your site.

Who should use this document

This document should be used by those who plan, install, customize, administer, and use PKI Services. It should also be used by those who install, configure, or provide support in the following areas:

- Lightweight Directory Access Protocol (LDAP)
- Resource Access Control Facility (RACF®)
- z/OS
- z/OS UNIX System Services
- Integrated Cryptographic Service Facility (ICSF)

The following areas where this document should be used are optional:

- IBM® HTTP Server - Powered by Apache
- Open Cryptographic Enhanced Plug-ins (OCEP)
- Open Cryptographic Services Facility (OCSF)
- z/OS Communications Server's sendmail utility
- WebSphere Application Server

This document assumes that you have experience with installing and configuring products in a network environment. You should be knowledgeable about the following concepts and protocols:

- Hardware installation and configuration
- Internet communications protocols, in particular Transmission Control Protocol/Internet Protocol (TCP/IP) and Secure Sockets Layer (SSL)
- Public key infrastructure (PKI) technology, including directory schemas, the X.509 version 3 standard, and the Lightweight Directory Access Protocol (LDAP)

How to use this document

This document contains several parts:

- Part 1, “Planning,” on page 1 includes the following topics:
  - Chapter 1, “Introducing PKI Services,” on page 3 introduces PKI Services, describing its basic components and related products. It also describes supported standards, certificate types, fields, and extensions.

  - Chapter 2, “Planning your implementation,” on page 11 provides a planning overview for your implementation. It contains information about the components that work with PKI Services and the team members you need to implement PKI Services and the skills they need.
  - Chapter 3, “Installing and configuring prerequisite products,” on page 29 describes installing and configuring related products: the IBM HTTP Server - Powered by Apache, OCSF, LDAP, and ICSF.
- Part 2, “Configuring your system for PKI Services,” on page 37 describes the tasks your team members need to perform to configure PKI Services.
  - Chapter 4, “Running IKYSETUP to perform RACF administration,” on page 39 describes how the RACF administrator updates and runs IKYSETUP, a REXX exec to perform RACF administration tasks, such as setting up the daemon user ID and giving accesses.
  - Chapter 5, “Configuring the UNIX runtime environment,” on page 67 explains UNIX programmer tasks including how to copy files, update environment variables, update the PKI Services configuration file, and set up the `/var/pkiserv` file system directory.
  - Chapter 6, “Tailoring the LDAP configuration for PKI Services,” on page 97 explains how the LDAP programmer updates LDAP configuration for PKI Services.
  - Chapter 7, “Updating IBM HTTP Server - Powered by Apache configuration and starting the server,” on page 101 explains how the web server programmer updates the IBM HTTP Server configuration files and starts the IBM HTTP Server.
  - Chapter 8, “Tailoring the PKI Services configuration file for LDAP,” on page 107 explains how the UNIX programmer updates the **LDAP** section of the PKI Services configuration file.
  - Chapter 9, “Creating the object store and ICL,” on page 115 explains how the MVS™ programmer creates VSAM data sets or the Db2 database administrator creates Db2 objects for the object store and issued certificate list (ICL).
  - Chapter 10, “Starting and stopping PKI Services,” on page 129 explains how the MVS programmer starts and stops the PKI Services daemon.
- Part 3, “Customizing PKI Services,” on page 133 explains how to customize end-user and administration web pages and perform advanced customization, including customization using an exit routine.
  - Chapter 11, “Customizing the end-user web application if you use REXX CGI execs,” on page 135 provides an overview of the `pkiserv.tmpl` file, which contains the certificate templates, and explains how to customize the end-user web pages.
  - Chapter 12, “Customizing the administration web pages if you use REXX CGI execs,” on page 233 provides an overview of the CGI scripts and explains how to customize the administration web pages.
  - Chapter 13, “Implementing the web application using JavaServer pages,” on page 237 describes how to use Java™ server pages (JSPs) and an XML template file to create and customize the PKI Services web application.
  - Chapter 14, “Advanced customization,” on page 283 explains how to use certificate policies, update the signature algorithm, enable automatic renewal of certificates, set up PKI Services to generate keys for certificates, and add custom extensions to certificates.
  - Chapter 17, “Customizing with installation exit routines,” on page 349 explains how to use the PKI Services exit routine.
- Part 4, “Using PKI Services,” on page 373 explains using the end-user and administration web pages, using the PKI Services utilities, and using the certificate management protocol (CMP) with PKI Services.

  Chapter 18, “Using the end-user web pages,” on page 375 shows the end-user web pages and explains how to request a certificate, obtain the certificate, and renew or revoke a certificate.

  Chapter 19, “Using the administration web pages,” on page 405 shows the administration web pages and explains how to process certificate requests and certificates.

  Chapter 20, “Using PKI Services utilities,” on page 431 explains using the PKI Services utilities.

  Chapter 21, “Using the certificate management protocol (CMP) with PKI Services,” on page 453 describes the support for CMP that PKI Services provides.

- Part 5, "Administering security for PKI Services," on page 479 explains how to perform many RACF administration tasks that are needed for PKI Services, such as authorizing users, administering extensions, and locating your PKI Services certificate and key ring.
- Part 7, "Troubleshooting," on page 527 explains using logs:
  - Chapter 24, "Using information from SYS1.LOGREC," on page 529 describes SYS1.LOGREC - which is used to record unusual runtime events, such as an exception.
  - Chapter 25, "Using information from the PKI Services logs," on page 537 contains information about using the PKI Services logs to debug problems and explains how to change logging options and display log options settings.
- Part 8, "Reference information," on page 543 provides reference information including messages and important code samples.
  - Chapter 26, "Messages," on page 547 explains PKI Services messages.
  - Chapter 27, "File directory structure," on page 603 describes product and file system directories for PKI Services and files that are contained in them.
  - Chapter 28, "The pkiserv.conf configuration file," on page 607 provides a code sample of the `pkiserv.conf` configuration file.
  - Chapter 29, "Environment variables," on page 615 explains the `pkiserv.envars` environment variables file and provides a code sample.
  - Chapter 30, "The IKYSETUP REXX exec," on page 619 explains the contents of the IKYSETUP REXX exec that performs RACF administration and provides a code sample.
  - Chapter 31, "Other code samples," on page 649 provides additional code samples.
  - Chapter 32, "SMF recording," on page 683 describes the SMF record that PKI Services produces.
- There are several appendixes, including:
  - Appendix A, "LDAP directory server requirements," on page 685 explains using a non-z/OS LDAP server.
  - Appendix B, "Using a gskkyman key database for your certificate store," on page 689 explains an alternative method for setting up your key database.
  - Appendix C, "Using the PKI Services web application with Internet Explorer on Windows systems," on page 691 describes how you might need to set up a Windows system to work with PKI Services.

z/OS information

This information explains how z/OS references information in other documents and on the web.

When possible, this information uses cross document links that go directly to the topic in reference using shortened versions of the document title. For complete titles and order numbers of the documents for all products that are part of z/OS, see *z/OS Information Roadmap*.

To find the complete z/OS library, go to IBM Knowledge Center (www.ibm.com/support/knowledgecenter/SSLTBW/welcome).

How to send your comments to IBM

We invite you to submit comments about the z/OS product documentation. Your valuable feedback helps to ensure accurate and high-quality information.

Important: If your comment regards a technical question or problem, see instead "If you have a technical problem" on page xxvii.

Submit your feedback by using the appropriate method for your type of comment or question:

Feedback on z/OS function
If your comment or question is about z/OS itself, submit a request through the IBM RFE Community (www.ibm.com/developerworks/rfe/).

Feedback on IBM Knowledge Center function
If your comment or question is about the IBM Knowledge Center functionality, for example search capabilities or how to arrange the browser view, send a detailed email to IBM Knowledge Center Support at ibmkc@us.ibm.com.

Feedback on the z/OS product documentation and content
If your comment is about the information that is provided in the z/OS product documentation library, send a detailed email to mhvrcfs@us.ibm.com. We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information.

To help us better process your submission, include the following information:

- Your name, company/university/institution name, and email address
- The following deliverable title and order number: z/OS Cryptographic Services PKI Services Guide and Reference, SA23-2286-40
- The section title of the specific information to which your comment relates
- The text of your comment.

When you send comments to IBM, you grant IBM a nonexclusive authority to use or distribute the comments in any way appropriate without incurring any obligation to you.

IBM or any other organizations use the personal information that you supply to contact you only about the issues that you submit.

If you have a technical problem

If you have a technical problem or question, do not use the feedback methods that are provided for sending documentation comments. Instead, take one or more of the following actions:

- Go to the IBM Support Portal (support.ibm.com).
- Contact your IBM service representative.
- Call IBM technical support.

Summary of changes

This information includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations for the current edition are indicated by a vertical line to the left of the change.

Summary of changes for z/OS Version 2 Release 4 (V2R4)

The following changes are made to z/OS Version 2 Release 4 (V2R4).

The most recent updates are listed at the top of each section.

New

The following information is new:

August 2020 refresh

New statements have been added to and existing statements have been removed from the configuration file for vhost80.conf, vhost443.conf, and vhost1443.conf, see "IBM HTTP Server - Powered by Apache (optional)" on page 13.

Prior to August 2020 refresh

- A new topic has been added for Using Enrollment over Secure Transport (EST). See Chapter 16, "Using Enrollment over Secure Transport (EST)," on page 341 for more information. Additional updates for EST include:
  - A new EST entry has been added to the "Supported standards" on page 6 topic.
  - A new template for EST requests has been added to `pkiserv.tmpl`, see "Examining the CUSTOMERS application" on page 168, "Locating code for customizing end-user web pages" on page 224, and "Examining the TEMPLATE section" on page 179.
  - New statements have been added to and existing statements have been removed from the configuration file for `vhost80.conf`, `vhost443.conf`, and `vhost1443.conf` see "IBM HTTP Server - Powered by Apache configuration directives" on page 649.
  - `SYS1.SAMPLIB` has been updated to add Subject Alternate Name (SAN) for generation of root EST CA certificate and web server certificate. See "IKYSETUP sample" on page 624.
  - New entries have been added for the 2-Year EST Certificate - Preregistration. See "TEMPLATE sections" on page 151.
  - The description for "Examining the pkitmpl.xml file" on page 237 has been updated with EST updates.
  - The request page's dropdown box for extended key usage has been updated to include five new values and new values have been added to the extended keyusage INSERT in `pkiserv.tmpl`. See "Examining the INSERT section" on page 185.
  - New key values have been added to the entry for Extended key usage in the Summary of fields in end-user pages table. See "Summary of fields" on page 379.
  - The description of the `_PKISERV_CMP_EXTKEYUSAGE_domain` in the HTTP Server environment variables table has been updated. See "Setting up the HTTP Server for CMP" on page 465.
  - New keywords EnableEST, ESTCAFile and ESTTEmplate under the CertPolicy section of the pkiserv.conf. See Chapter 28, "The pkiserv.conf configuration file," on page 607.
- A new topic has been added for setting up PKI Services to generate certificates synchronously when requesting certificates through the web interfaces. See "Setting up synchronous certificate requests" on page 325. For additional information about fulfilling synchronous certificates, see the following topics:

    - A new paragraph describing the SYNCHRONOUS tag has been added to "TEMPLATE sections" on page 151.
    - Information about the SYNCHRONOUS tag has been added to "Examining the TEMPLATE section" on page 179 and "Examining the pkitmpl.xml file" on page 237.
  - Three RSAPSS signing algorithms are now supported by PKI Services. See "Updating the signature algorithm" on page 287 and Chapter 28, "The pkiserv.conf configuration file," on page 607 for more information:

- New Messages:

  IKYC098I
  IKYP055I
  IKYP056I

Changed

The following information has been changed:

August 2020 refresh

  - The procedure for setting up the key ring for Liberty users to access the PKI Services web pages has been updated, see "Steps for setting up the key ring" on page 271.
  - The steps for accessing the end-user web pages has been updated, see "Steps for accessing the end-user web pages" on page 376.
  - The steps for access the administration home page has been updated, see "Steps for accessing the administration home page" on page 405.
  - The procedure for installing the PKI Services CA certificate on a Microsoft Windows system has been updated , see "Steps for installing the PKI Services CA certificate on a Microsoft Windows system" on page 695.
  - The configuration files for `vhost80.conf`, `vhost443.conf`, and `vhost1443.conf`have been updated, see "IBM HTTP Server - Powered by Apache configuration directives" on page 649.

- Changed Messages:

  IKYC055I
  IKYC056I
  IKYC058I
  IKYP031E

Deleted

- None
- Deleted Messages:

  None

Summary of changes for z/OS Version 2 Release 3 (V2R3)

The following changes are made to z/OS Version 2 Release 3 (V2R3).

The most recent updates are listed at the top of each section.

New

- New alternative to WebSphere Application Server, WebSphere Liberty (optional), added to "Basic components of PKI Services and related products" on page 4 and "Component diagram" on page 5.
- New section added for "Requirements for FIPS" on page 12.

- “WebSphere Application Server / z/OS Liberty Embedded (optional)” on page 13 is updated to include new alternative, z/OS Liberty Embedded.
- New skill requirements identified in “Skills for setting up prerequisite products” on page 15 for Liberty.
- New step added to “(Optional) Steps for updating PKI Services environment variables” on page 72 to include `_PKISERV_FIPS_LEVEL`.
- New parameters are added to Table 21 on page 74 and Chapter 28, “The pkiserv.conf configuration file,” on page 607:
  - `DBVersion`
  - `DBWaitTime`
  - `ObjectSCEPTidDSN`
  - `ICLSCEPTidDSN`
- New section added, “Sysplex considerations” on page 116, for VSAM.
- New task added for creating multiple applications using Traditional WebSphere Application Server, “Steps for creating multiple applications” on page 260.
- New task, “Steps for creating a new EAR alongside your existing EAR” on page 260, is added.
- New task added for giving WebSphere users authorization to use PKI Services functions in Liberty, “Steps for giving Liberty users authorization to use PKI Services functions” on page 268.
- New task added for allowing Liberty users to set up the key ring, “Steps for setting up the key ring” on page 271.
- New sections added for SCEP requester enhancement, “Overview of SCEP requester enhancement” on page 333, “Steps for converting existing VSAM object store and ICL to the new format” on page 333, and “Steps for converting existing Db2 object store and ICL to the new format” on page 333.
- New conversion utilities added, “Using the db2conv utility” on page 432 and “Using the vsamconv utility” on page 450.
- New table added, HTTP Server environment variables used to control the FIPS level, within “Setting up the HTTP Server for CMP” on page 465.
- New CMP error codes added in “Messages and codes returned from the CMP functions” on page 473.
- New procedure added, “Steps to retire and replace the PKI Services non-FIPS compliant CA private key for the PKI templates” on page 493.
- New bullet for, CBRDBRM/ACBRDBRM, added to “Product libraries” on page 603.
- New keyword, `_PKISERV_FIPS_LEVEL`, added to “The pkiserv.envars environment variables file” on page 617.
- New code sample, `server.xml`, added in “WebSphere Liberty” on page 653.
- New sample jobs “IKYCDBV1” on page 658, “IKYCVSV1” on page 665, and “IKYRVSV1” on page 673 are added and reflected in Table 103 on page 543 and “Product libraries” on page 603.
- Messages:

  IKYC091I
  IKYC092I
  IKYC093I
  IKYC094I
  IKYC095I
  IKYC096I
  IKYC097I
  IKYC904I
  IKYC905I
  IKYO007I
  IKYP047I
  IKYP048I

```
IKYP049I
IKYP050I
IKYP051I
IKYP052I
IKYP053I
IKYP054I
IKYU019I
```

Changed

- "Steps for establishing your CA and RA certificates" on page 622 is updated to include RDATALIB support.
- The following updates are in support of the Communications Server `sendmail` change in Version 2 Release 3:
  - "Configuring sendmail (optional)" on page 34
  - "Optionally updating PKI Services environment variables" on page 70
  - "(Optional) Steps for updating PKI Services environment variables" on page 72
  - "The pkiserv.envars environment variables file" on page 617
- The use of ICSF is required for PKI Services as reflected in "Basic components of PKI Services and related products" on page 4.
- Chapter 2, "Planning your implementation," on page 11, "ICSF" on page 14, and "Planning considerations for installing and configuring PKI Services" on page 19 are updated to reflect ICSF changes and FIPS mode support.
- Note added to "Deciding the value of key_type" on page 45 for FIPS compliance.
- "Table of IKYSETUP variables that you might want to change" on page 49 updated to include FIPS support.
- Note added to "The object store and ICL" on page 115 to include SCEP requester enhancement support.
- The following are updated to reflect VSAM changes due to SCEP requester enhancement:
  - "Planning VSAM storage requirements" on page 116
  - "(Optional) steps for enabling existing PKI Services VSAM data sets for VSAM RLS" on page 119
  - "Tuning VSAM performance" on page 120
- The following are updated to reflect Db2 changes due to SCEP requester enhancement:
  - "Creating the object store and ICL using Db2 tables" on page 122
  - "Sysplex considerations" on page 123
  - "Planning Db2 storage requirements" on page 123
  - "Steps for creating the object store and ICL Db2 tables" on page 123
  - "Steps for converting the object store and ICL from VSAM to Db2" on page 125
  - "Columns in the ICL and object store Db2 tables" on page 126
- "Using the iclview utility" on page 433 and "Using the vsamconv utility" on page 450 are updated to include new parameters for the version specification of a Db2 package.
- "Using the vsam2db2 utility" on page 448 and Chapter 28, "The pkiserv.conf configuration file," on page 607 descriptions are updated to include new keywords and options.
- "IKYSETUP sample" on page 624, "IKYSBIND" on page 677, and "IKYSGRNT" on page 679 are updated to include new sample jobs.
- The following information is updated to include modifications for the removal of support for PKIXEnroll:
  - "Examining the APPLICATION section" on page 166

  - "Examining the PKISERV application" on page 166
  - "Examining the CUSTOMERS application" on page 168
  - "Examining the INSERT section" on page 185
  - "Examining the pkitmpl.xml file" on page 237
  - "File system directory and subdirectories" on page 603
  - The sample configuration directives in "IBM HTTP Server - Powered by Apache configuration directives" on page 649
  - "Administrator tasks for setting up a Windows system and Internet Explorer to work with the PKI Services web application" on page 695
  - "Steps for signing the PKI Services ActiveX programs" on page 696
  - "Steps for building the installer programs using Microsoft Visual Studio" on page 697
- Messages:

  IKYP003I
  IKYP025I
  IKYP040I

Deleted

- Appendix C. Configuring PKI Service as an Identrust certificate authority is deleted.
- Messages:

  None

z/OS Version 2 Release 2 (V2R2) summary of changes as updated March 2017

The following changes are made to z/OS Version 2 Release 2 (V2R2) as updated March 2017.

New

- The parameter, `CRLEnhancements` is added to Table 21 on page 74.

Changed

- The **Information needed** section in Table 21 on page 74 is updated for the parameter, `CRLDuration`.
- The **Information needed** section in Table 21 on page 74 is updated for the parameter, `EnableLargeCRLPosting`.
- A note is added to "Enabling support for large CRLs" on page 297.
- Chapter 28, "The pkiserv.conf configuration file," on page 607 is updated to include the new parameter, `CRLEnhancements`.

Summary of changes for z/OS Version 2 Release 2 (V2R2)

The following changes are made to z/OS Version 2 Release 2 (V2R2).

New

- "Planning considerations for installing and configuring PKI Services" on page 19 information is added.
- Variables `AdminNotifyModForm` and `PKCS12Content` are added to Table 21 on page 74.

- New Inserts ObjectHeaderIEXP and ObjectHeaderIENONXP are added to Table 31 on page 138. For more information, see "INSERT sections" on page 137, "What are substitution variables?" on page 136, and "Examining the INSERT section" on page 185.
- New template keyword ADMINNUM, has been added in "TEMPLATE sections" on page 151 and "Examining the pkitmpl.xml file" on page 237.
- "Customizing email notifications sent to users" on page 317 is updated to add the `%%modreqlist%%` variable.
- Signing algorithms for DSA are added to Chapter 28, "The pkiserv.conf configuration file," on page 607 and "Updating the signature algorithm" on page 287.
- Form `pendingmsg2` is added to Chapter 5, "Configuring the UNIX runtime environment," on page 67, "Steps for copying files" on page 69, "(Optional) Steps for updating the configuration file" on page 74, "Customizing email notifications sent to users" on page 317, "File system directory and subdirectories" on page 603, and Chapter 28, "The pkiserv.conf configuration file," on page 607.
- Module IRRSPX64 is added. For more information, see "Basic components of PKI Services and related products" on page 4.
- "Locating JSP files for customizing web pages" on page 276 and "Examining the INSERT section" on page 185are updated because VBScript is no longer used.
- "Steps for accessing the end-user web pages" on page 376, "Steps for renewing a certificate" on page 395, "User tasks for setting up a Windows system and Internet Explorer to work with the PKI Services web application " on page 691, "Installing the PKI Services ActiveX program" on page 691, "Steps for installing the PKI Services ActiveX program when you renew a certificate" on page 693, "Administrator tasks for setting up a Windows system and Internet Explorer to work with the PKI Services web application" on page 695, and Appendix C, "Using the PKI Services web application with Internet Explorer on Windows systems," on page 691 are updated because CAPICOM is no longer supported.
- Messages:
  - IKYC090I

Changed

- IBM HTTP Server 5.3 is no longer available. IBM HTTP Server - Powered by Apache is used, and updates can be found in these topics:

  "Basic components of PKI Services and related products" on page 4
  "IBM HTTP Server - Powered by Apache (optional)" on page 13
  Chapter 7, "Updating IBM HTTP Server - Powered by Apache configuration and starting the server," on page 101
  "IBM HTTP Server - Powered by Apache configuration directives" on page 649
- The `pkiserv.tmpl` file is updated in the following sections:

  "Examining the PKISERV application" on page 166
  "Examining the CUSTOMERS application" on page 168
  "Examining the TEMPLATE section" on page 179
  "Examining the INSERT section" on page 185
- The Native Library Path field in "Steps for deploying the EAR file to a Traditional WebSphere Application Server" on page 261 is changed.
- The TemplateTool utility is updated support to handle multiple approvers.

Part 1. Planning

The Planning part includes the following topics:

- Chapter 1, “Introducing PKI Services,” on page 3 provides an overview of PKI Services, its components, and related concepts.
- Chapter 2, “Planning your implementation,” on page 11 provides a planning overview for your implementation, including information about the components that work with PKI Services. It also contains information about the team members you need to implement PKI Services and the skills they needed.
- Chapter 3, “Installing and configuring prerequisite products,” on page 29 describes installing and configuring related products: the IBM HTTP Server - Powered by Apache, LDAP, ICSF and OCSF.

Chapter 1. Introducing PKI Services

This topic provides an overview of PKI Services.

It covers the following topics:

What is PKI Services?

z/OS Cryptographic Services PKI Services allows you to use z/OS to establish a PKI infrastructure and serve as a certificate authority for your internal and external users, issuing and administering digital certificates in accordance with your own organization's policies. Your users can use a PKI Services application to request and obtain certificates through their own web browsers, while your authorized PKI administrators approve, modify, or reject these requests through their own web browsers. The web applications provided with PKI Services are highly customizable, and a programming exit is also included for advanced customization. You can allow automatic approval for certificate requests from certain users and, to provide additional authentication, add host IDs, such as RACF user IDs, to certificates you issue for certain users. You can also issue your own certificates for browsers, servers, and other purposes, such as virtual private network (VPN) devices, smart cards, and secure email.

PKI Services supports Public Key Infrastructure for X.509 version 3 (PKIX) and Common Data Security Architecture (CDSA) cryptographic standards. It also supports the following functions:

- The delivery of certificates through the Secure Sockets Layer (SSL) for use with applications that are accessed from a web browser or web server.
- The delivery of certificates that support the Internet Protocol Security standard (IPSEC) for use with secure VPN applications or IPSEC-enabled devices.
- The delivery of certificates that support Secure Multipurpose Internet Mail Extensions (S/MIME), for use with secure email applications.

What is a certificate authority?

The certificate authority, commonly called a CA, acts as a trusted third party to ensure that users who engage in e-business can trust each other. A certificate authority vouches for the identity of each party through the certificates it issues. In addition to proving the identity of the user, each certificate includes a public key that enables the user to verify and encrypt communications.

The trustworthiness of the parties depends on the trust that is placed in the CA that issued the certificates. To ensure the integrity of a certificate, the CA digitally signs the certificate as part of creating it, using its signing private key. Trying to alter a certificate invalidates the signature and renders it unusable.

Protecting the CA's signing private key is critical to the integrity of the CA. For this reason, you should consider using ICSF to securely store your PKI Services CA's private key.

As a CA using PKI Services, you can do the following tasks:

- Track certificates you issue with an issued certificate list (ICL) that contains a copy of each certificate, indexed by serial number
- Track revoked certificates using certificate revocation lists (CRLs). When a certificate is revoked, PKI Services updates the CRL during the next periodic update. Just as it signs certificates, the CA digitally signs all CRLs to vouch for their integrity.

What is PKI?

The public key infrastructure (PKI) provides applications with a framework for performing the following types of security-related activities:

- Authenticate all parties that engage in electronic transactions
- Authorize access to sensitive systems and repositories
- Verify the author of each message through its digital signature
- Encrypt the content of all communications.

The PKIX standard evolved from PKI to support the interoperability of applications that engage in e-business. Its main advantage is that it enables organizations to conduct secure electronic transactions without regard for operating platform or application software package.

The PKIX implementation in PKI Services is based on the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, RFC 5280 (tools.ietf.org/html/rfc5280).

Basic components of PKI Services and related products

Table 1. Basic components of PKI Services and related products

Component	**Description**
Administration web application	Assists authorized administrators to review requests for certificates, approve or reject requests, renew certificates, or revoke certificates through their own web browsers. The application consists of sample screens that you can easily customize to display your organization's logo. It also supports the following tasks: • Reviewing pending certificate requests • Querying pending requests to process those that meet certain criteria • Displaying detailed information about a certificate or request • Monitoring certificate information, such as validity period • Annotating the reason for an administrative action
Db2 (optional)	Provides an alternative to VSAM data sets as the repository for the object store (request database) and issued certificate list (ICL).
End-user web application	Guides your users to request, obtain, and renew certificates through their web browsers. The application consists of sample screens that you can easily customize to meet your organization's needs for certificate content and standards for appearance. It offers several certificate templates that you can use to create requests for various certificate types, based on the certificate's intended purpose and validity period, and supports certificate requests that are automatically approved.

Table 1. Basic components of PKI Services and related products (continued)

Component	**Description**
Exit	Provides advanced customization including additional authorization checking, additional validation, changes to parameters on calls to the `R_PKIServ` callable service (IRRSPX00 and IRRSPX64), and capture of certificates for further processing. An exit program can be called from the daemon, for automatic certificate renewal, or from the `PKIServ` CGIs. Exit methods can be called from JavaServer pages (JSPs). The exit program and methods support both preprocessing and post-processing functions. A code sample in C language code is included.
IBM HTTP Server (optional)	PKI Services uses the web server to encrypt messages, authenticate requests, and transfer certificates to intended recipients if you implement the PKI Services web application using REXX CGI scripts. You can use IBM HTTP Server - Powered by Apache as the web server.
ICSF	Provides cryptographic functions including key generation and stronger protection of keys generated by PKI Services and the CA key.
LDAP	The directory that maintains information about the valid and revoked certificates that PKI Services issues in an LDAP-compliant format. You can use an LDAP server such as the one provided by IBM Tivoli® Directory Server for z/OS.
PKI Services daemon	The server daemon that acts as your certificate authority, confirming the identities of users and servers, verifying that they are entitled to certificates with the requested attributes, and approving and rejecting requests to issue and renew certificates. It includes support for: • An issued certificate list (ICL) to track issued certificates • Certificate revocation lists (CRLs) to track revoked certificates
`R_PKIServ` callable service (IRRSPX00 and IRRSPX64)	The application programming interface (API) that allows authorized applications, such as servers, to programmatically request the functions of PKI Services to generate, retrieve and administer certificates.
RACF (or equivalent)	Protects the components of your PKI Services system by controlling who can use the functions of the `R_PKIServ` callable service. The RACF (or equivalent) database is where the certificate authority's certificate, private key (or private key label, if ICSF is chosen to store the private key) and key ring are stored. RACF (or equivalent) can also be used to create the certificate authority's certificate and key directly.
WebSphere Application Server (optional)	Serves as the application server if you implement the PKI Services web application using JavaServer pages (JSPs).
z/OS Liberty Embedded (optional)	An alternative to WebSphere Application Server to implement PKI Services web application using JavaServer pages (JSPs).

Component diagram

Figure 1 on page 6 shows a typical PKI Services system.

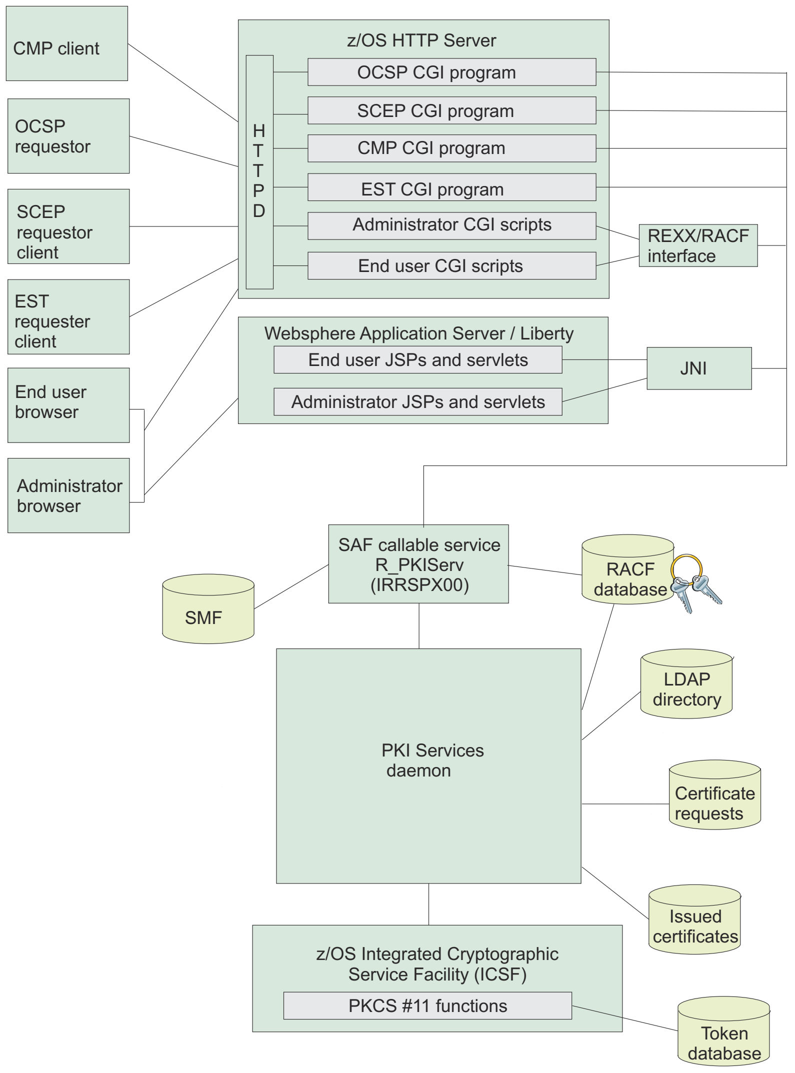


Figure 1. Component diagram of a typical PKI Services system

Supported standards

PKI Services supports the following standards for public key cryptography:

- Secure Sockets Layer (SSL) version 2 and version 3, with client authentication
- PKCS #10 browser and server certificate format, with a base64-encoded response
- IPSEC certificate format
- S/MIME certificate format
- Browser certificates for:
  - 32-bit versions of Microsoft Internet Explorer
  - Mozilla-based browsers such as Mozilla Firefox
- Server certificates
- LDAP standard for communications with the directory
- X.509v3 certificates
- Certificate revocation lists (CRLv2)
- Key lengths up to 4096 bits for the RSA CA signing private keys and up to 1024 bits for DSA keys
- RSA algorithms for encryption and signing

- DSA algorithms for signing
- ECC algorithms for encryption and signing
- RFC 2560 (tools.ietf.org/html/rfc2560)
- RFC 4291 (tools.ietf.org/html/rfc4291)
- RFC 4210 (tools.ietf.org/html/rfc4210)
- RFC 4211 (tools.ietf.org/html/rfc4211)
- RFC 6277 (tools.ietf.org/html/rfc6277)
- RFC7030 (tools.ietf.org/html/rfc7030)
- Cisco Systems' Simple Certificate Enrollment Protocol (SCEP)

The LDAP standard that PKI Services supports is LDAP Version 2. A directory using LDAP Version 3 (with RFC 1779 (tools.ietf.org/html/rfc1779) syntax) is acceptable if it is backwardly compatible with Version 2.

PKI Services supports the RDNs needed for an Extended Validation (EV) certificate. The criteria for issuing EV certificates are defined by the Extended Validation Certificate Guidelines (cabforum.org/extended-validation) produced by the CA/Browser Forum. PKI Services does not enforce these criteria. If you want to issue EV certificates, it is your responsibility to enforce the criteria.

Supported certificate types

Table 2 on page 7 lists the types of certificates that you can request, based on the certificate templates that are included with PKI Services. Certificate templates are samples of the most commonly requested certificate types. You can add, modify, and remove certificate templates to customize the variety of certificate types you offer to your users.

Table 2. Types of certificates you can request

Type of certificate	**Use**
One-year PKI SSL browser certificate	End-user client authentication using SSL
One-year PKI S/MIME browser certificate	Browser-based email encryption
One-year PKI generated key certificate	Generation of public and private keys by PKI Services
Two-year EV SSL server certificate	SSL web server certification with verification of the subject’s identity meeting the guidelines for extended validation (EV) certificates.
Two-year PKI browser certificate for authenticating to z/OS	End-user client authorization using SSL when logging on to z/OS
Two-year PKI Authenticode - code signing server certificate	Software signing
Two-year PKI Windows logon certificate	End-user client authentication for an Active Directory user logging in to a Windows desktop using a smart card
Five-year PKI SSL server certificate	SSL web server certification
Five-year PKI IPSEC server (firewall) certificate	Firewall server identification and key exchange
Five-year PKI intermediate CA server certificate	Subordinate (non-self-signed) certificate authority certification

Table 2. Types of certificates you can request (continued)

Type of certificate	Use
Five-year SCEP certificate	Creation of a preregistration record for certificate requestors. (Certificate requestors using Simple Certificate Enrollment Protocol (SCEP) must be preregistered.) Unlike other templates, this template is intended for administration use only.
Two-year EST certificate	Preregister a user or device for an EST certificate and for EST certificate fulfillment.
n-year PKI browser certificate for extensions demonstration	Demonstration of all extensions supported by PKI Services
One-year SAF browser certificate	End-user client authentication where the security product (RACF, not PKI Services) is the certificate provider **Note:** The certificate generated by this template cannot be managed by the PKI Services administrator.
One-year SAF server certificate	Web server SSL certification where the security product (RACF, not PKI Services) is the certificate provider **Note:** The certificate generated by this template cannot be managed by the PKI Services administrator.

Note: You can customize certificate templates to add, modify, and remove certificate types.

Supported certificate fields and extensions

PKI Services certificates support most of the fields and extensions defined in the X.509 version 3 (X.509v3) standard. This support lets you use these certificates for most cryptographic purposes, such as SSL, IPSEC, VPN, and S/MIME.

PKI Services supports Basic Latin and Latin-1 supplement characters in the Subject Distinguished Name, Issuer Distinguished Name and Othername in Subject Alternate Name.

PKI Services certificates can include the following types of extensions:

Standard extensions

The standard X.509v3 certificate extensions:

- authority information access
- authority key identifier
- basic constraints
- certificate policies
- certificate revocation list (CRL) distribution points
- extended key usage
- key usage
- subject alternate name
- subject key identifier

Other extensions
Extensions that are unique to PKI Services, such as host identity mapping. This extension associates the subject of a certificate with a corresponding identity on a host system, such as with a RACF user ID.

To support your organization's policies, PKI Services provides the means for you to select and customize the supported certificate extensions. For example, you can change the extensions that are specified in the default certificate templates or create templates that return certificates with different extensions. In addition, you can include your own extensions in your certificates by defining custom extensions.

Chapter 2. Planning your implementation

The implementation of PKI Services requires the interaction of several software products, each with its own required skills. Therefore, it is important to understand the tasks involved and to plan your implementation.

This topic provides the information you need to understand the task of implementing PKI Services, determine which skills are required to complete your implementation team, and create your own implementation plan.

This topic covers the following topics:

- "Installing PKI Services" on page 11
- "Requirements for sysplex support" on page 11
- "Requirements for FIPS" on page 12
- "Determining prerequisite products" on page 12
- "Identifying skill requirements" on page 14
- "Creating an implementation plan" on page 19.

Installing PKI Services

Your MVS programmer uses SMP/E to install PKI Services into a file system directory. By default, PKI Services is installed in the `/usr/lpp/pkiserv` directory but the MVS programmer can determine whether to change the default for this and other directories. Before your team begins installing and configuring prerequisite products and setting up PKI Services, you need to know which file system directories were used so you can customize the installation process.

Table 3 on page 11 shows each file system variable with its description and default value. Your MVS programmer should review the rightmost column of this table, crossing out any defaults that have changed and recording the correct directory names.

Table 3. File system directory variables

Variable name	Description	Default value or customized value
variables-dir	The file system directory where PKI Services creates working files.	`/var/pkiserv`
install-dir	The file system directory where PKI Services is installed.	`/usr/lpp/pkiserv`
runtime-dir	The file system directory where PKI Services looks for configuration files.	`/etc/pkiserv`

Requirements for sysplex support

If your installation plans to use sysplex support (running multiple independent instances of PKI Services, one per image, that work in unison):

- If you are using VSAM data sets for the object store and ICL, all systems in the sysplex that run PKI Services must have the same release of z/OS installed, and all instances of PKI Services must share the same VSAM data sets. To do so, they use VSAM record-level sharing (RLS). This requires setting up a coupling facility for data sharing (lock and cache).
- If you are using Db2 tables for the object store and ICL, all systems in the sysplex that run PKI Services must be at z/OS V1R13 or later and have the same release of z/OS installed. Db2 Version 9 or later must

be installed and the Db2 subsystems that share data must belong to a Db2 data sharing group that runs on a sysplex cluster.

See "(Optional) preliminary steps for establishing VSAM RLS" on page 117 for information about creating VSAM data sets suitable for VSAM RLS. For information about establishing a Parallel Sysplex® environment with a coupling facility, see *z/OS MVS Programming: Sysplex Services Guide*. For more information about establishing data sharing for VSAM RLS, see *z/OS DFSMS Introduction* and *z/OS DFSMSdfp Storage Administration*.

Requirements for FIPS

PKI Services provides the capability to execute securely to meet the National Institute Standards and Technology (NIST) Federal Information Processing Standards (FIPS) levels supported by System SSL and ICSF PKCS#11. System SSL supports 3 FIPS levels. Each level has different requirements on key size and signature algorithms. The FIPS level to be used by PKI Services is specified by the environment variable _PKISERV_FIPS_LEVEL:

- Level 1 corresponds to FIPS 140-2.
- Level 2 corresponds to SP800-131-A with exception.
- Level 3 corresponds to SP800-131-A without exception.

Requirements for the CA certificate

Based on the requirements from System SSL, the CA certificate must meet the following requirements:

- The key must not be stored in ICSF Public Key Data Set (PKDS)
- The key cannot be a Brainpool ECC key.
- For RSA and DSA key, key size must be at least 1024 bits (FIPS level 1) or 2048 bits (FIPS level 2 and 3); for NIST ECC key, key size must be at least 192 bits (FIPS level 1) or 224 (FIPS level 2 and 3).
- The hash algorithm used in the signature on the CA certificate must be SHA1 or higher (FIPS level 1 and 2) or SHA224 (FIPS level 3) or higher.

Note: If the CA certificate is a DSA certificate generated by RACF RACDCERT prior to z/OS V2R2 or V2R2 without PTF UA80493 installed, it is not FIPS compliant. You will need to perform the REKEY and ROLLOVER process if you want to enable PKI Services to run in FIPS mode using the DSA CA certificate. This process is discussed in the section "Retiring and replacing the PKI Services CA private key" on page 491.

Requirements for the RA certificate

When PKI Services is configured to provide SCEP processing and an RA certificate is specified, the RA certificate must meet the same requirements as a CA certificate using an RSA key for the desired FIPS level.

Requirements for the signing algorithm

The hash algorithm used by PKI Services to sign issued certificates and CRLs must be SHA or higher for FIPS level 1, or SHA224 or higher for FIPS level 2 and 3. This hash algorithm is specified by the `SigAlg1` configuration keyword, described in Table 21 on page 74.

Determining prerequisite products

The installation and use of PKI Services requires the following products:

- "IBM HTTP Server - Powered by Apache (optional)" on page 13
- "WebSphere Application Server / z/OS Liberty Embedded (optional)" on page 13

- “LDAP directory server” on page 13
- “OCSF” on page 13
- “ICSF” on page 14
- “sendmail (optional)” on page 14
- “OCEP (optional)” on page 14.
- “Db2 (optional)” on page 14.

The installation and use of RACF, or an equivalent security product, is required.

IBM HTTP Server - Powered by Apache (optional)

In a PKI Services system, if you implement the web application using REXX CGI execs the HTTP server handles all requests that it receives from a web browser. The requests can include requests for new certificates and requests to renew or revoke existing certificates. If needed, it performs authentication before allowing any exchange of information to take place. The IBM HTTP Server - Powered by Apache is required if you use the REXX CGI web application or if you use one or more of the CGI programs. CGI programs provide OCSP, SCEP, EST, and the CMP support for PKI Services.

PKI Services supports IBM HTTP Server - Powered by Apache. It is part of the IBM z/OS operating system. For more information, see IBM HTTP Server (www.ibm.com/software/products/http-servers). It is also included with WebSphere Application Server.

The HTTP server must be installed on the same system on which PKI Services is installed. SSL-enablement is required. If your HTTP server is SSL-enabled, your key file can be a RACF key ring, or a key file created by another product. For more information, see “Steps for installing and configuring the IBM HTTP Server - Powered by Apache to work with PKI Services ” on page 30.

WebSphere Application Server / z/OS Liberty Embedded (optional)

If you implement the PKI Services web application using JavaServer pages (JSPs), you must use WebSphere Application Server 7.0 or later or z/OS Liberty Embedded 8.5 or later. Liberty is included in the base level of z/OS Version 2 Release 3. An application server is a Java virtual machine (JVM) running user applications. WebSphere Application Server provides application servers that handle web application requests.

If you implement the PKI Services web application using REXX CGI execs, WebSphere Application Server or Liberty is not required.

LDAP directory server

Use of an LDAP server is required to maintain information about PKI Services certificates in a centralized location. The z/OS LDAP server provided by IBM Tivoli Directory Server for z/OS is preferred, but you can use a non-z/OS LDAP server if it can support the objectclasses and attributes that PKI Services uses. Typical PKI Services usage requires an LDAP directory server that supports the LDAP (Version 2) protocol (and the PKIX schema). If you use the z/OS LDAP server provided by IBM Tivoli Directory Server for z/OS, configure it for either the TDBM or LDBM backend.

Through the integration of the z/OS LDAP server with Db2, the directory can support millions of directory entries. It also allows client applications, such as PKI Services, to perform database storage, update, and retrieval transactions. For more information, see “Steps for installing and configuring LDAP” on page 31.

OCSF

You need to install and configure OCSF if your installation plans to write an application to implement the use of PKI Trust Policy (PKITP). For more information, see “Installing and configuring OCSF” on page 35.

ICSF

Beginning with z/OS V2R3, PKI Services relies on ICSF to provide basic cryptographic functions in addition to those that involve key generation and stronger protection of keys generated by PKI Services and the CA key. For more information, see "Installing and configuring ICSF" on page 33.

sendmail (optional)

You need to configure sendmail if your installation plans to send email notifications to users for certificate-related events, such as certificate expiration. For more information, see "Configuring sendmail (optional)" on page 34.

OCEP (optional)

You need to install and configure OCEP if your installation plans to write an application to implement the use of PKI Trust Policy (PKITP). For more information, see "Configuring and getting started with PKITP" on page 511.

Db2 (optional)

You need to install and configure Db2 if your installation chooses to implement the object store and issued certificate list (ICL) using Db2 instead of VSAM. For more information, see Chapter 9, "Creating the object store and ICL," on page 115.

Identifying skill requirements

The implementation of PKI Services requires the interaction of several software products, each with its own required skills. This means that your team might consist of people from several different disciplines, particularly if you work with a large organization.

This section provides the information that you need to determine which skills are required to complete your implementation. These skills are presented in terms of job titles for people who specialize in those skills. For example, a task requiring MVS skills is referred to as a task for an MVS programmer. Therefore, if some of your team members have multiple skills, you might require fewer individuals to complete your team.

Team members

Your team for installing and configuring prerequisite products and setting up PKI Services should include the following members:

- Db2 database administrator (DBA) (optional)
- ICSF programmer
- LDAP programmer
- MVS programmer
- OCEP programmer (optional)
- OCSF programmer (optional)
- RACF administrator
- UNIX programmer
- Web server programmer

You might want to include a web page designer to customize your PKI Services web applications. This task is listed in the topic as a task for a web server programmer.

One or more PKI administrators are needed to manage your ongoing operation as a certificate authority when your PKI Services system is set up. The responsibilities of these administrators include approving, modifying, and rejecting certificate requests and revoking certificates. It might be advisable to appoint a PKI administrator early, and involve this person in your planning.

Important: PKI administrators play a powerful role in your organization. The decisions that they make when managing certificates and certificate requests determine who accesses your computer systems and what privileges they have when doing so. Assign PKI administration duties to only highly trusted individuals.

Skills for setting up prerequisite products

The following table lists team members (alphabetically) and tasks and required skills needed for installing and configuring prerequisite products:

Table 4. Tasks and skills needed for installing prerequisite products

Role	Tasks	Required skills	Documented in:
Db2 database administrator (DBA)	(Optionally) installing and configuring Db2 (if not already done)	Db2 installation and configuration skills	• *Installation Guide* for your version of Db2.
ICSF programmer	Installing and configuring ICSF (if not already done)	ICSF installation and configuration skills	• *z/OS Cryptographic Services ICSF Administrator's Guide* • *z/OS Cryptographic Services ICSF Application Programmer's Guide* • *z/OS Cryptographic Services ICSF System Programmer's Guide*
	Setting up the ICSF PKCS #11 token data set (TKDS) (if not already done)	ICSF installation and configuration skills	• *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*
LDAP programmer	Installing and configuring LDAP (if not already done) and recording information	LDAP installation and configuration skills	• *z/OS IBM Tivoli Directory Server Administration and Use for z/OS*
OCEP programmer	(Optionally) Installing and configuring OCEP for use with PKITP	OCEP installation and configuration skills	• *z/OS Integrated Security Services Open Cryptographic Enhanced Plug-ins Application Programming*
OCSF programmer	(Optionally) Installing and configuring OCSF for use with PKITP	OCSF installation and configuration skills	• *z/OS Open Cryptographic Services Facility Application Programming*
UNIX programmer	(Optionally) Configuring sendmail if your installation is planning to send email notifications to users about certificates	• Basic UNIX commands such as the **cp** (copy) command and **mkdir** (make directory) command • sendmail configuration skills	• *z/OS Communications Server: IP Configuration Guide*

Table 4. Tasks and skills needed for installing prerequisite products (continued)

Role	Tasks	Required skills	Documented in:
Web server programmer	If you are implementing the PKI Services web application using REXX CGI execs, or if you plan to use OCSP, SCEP, EST, or CMP protocols: Installing and configuring the IBM HTTP Server - Powered by Apache (if not already configured for at least non-SSL pages) and recording information	IBM HTTP Server - Powered by Apache installation and configuration skills	• The WebSphere Application Server Knowledge Center (www.ibm.com/support/knowledgecenter/SSEQTP) for your release of WebSphere Application Server
	If you are implementing the PKI Services web application using JavaServer pages (JSPs): Installing and configuring WebSphere Application Server	WebSphere Application Server installation and configuration skills	• The WebSphere Application Server Knowledge Center (www.ibm.com/support/knowledgecenter/SSEQTP) for your release of WebSphere Application Server
	If you are implementing the PKI Services web application using JavaServer pages (JSPs) using Liberty: Installing and configuring Liberty	Liberty installation and configuration skills	• The WebSphere Application Server Knowledge Center (www.ibm.com/support/knowledgecenter/SSEQTP) for your release of WebSphere Application Server

Your team needs to install and configure prerequisite products before setting up PKI Services:

1. The web server programmer installs and configures the IBM HTTP Server - Powered by Apache (if you are implementing the PKI Services web application using REXX CGI execs or intend to use OCSP, SCEP, EST, or CMP protocols) or WebSphere Application Server (if you are implementing the PKI Services web application using JavaServer pages).
2. The LDAP programmer installs and configures LDAP.
3. The ICSF programmer installs and configures ICSF, and sets up the token data set (TKDS).
4. Optionally, the OCEP programmer installs and configures the OCEP.
5. Optionally, the OCSF programmer installs and configures the OCSF.
6. Optionally, the Db2 database administrator installs and configures Db2.

See Chapter 3, “Installing and configuring prerequisite products,” on page 29 for details about performing these tasks.

Skills for setting up PKI Services

The following table lists team members (alphabetically) and the tasks and skills needed for setting up PKI Services:

Table 5. Roles, tasks, and skills for setting up PKI Services

Role	Tasks	Required skills	Documented in:
Db2 database administrator	• Creates Db2 tables, package, and plan for the object store and issued certificate list (ICL).	• Db2 programming skills • Db2 administration skills	• *Application Programming and SQL Guide* • *Command Reference* • *SQL Reference* for your version of Db2.
LDAP programmer	• Customizes LDAP configuration for PKI Services	• LDAP customization skills	• *z/OS IBM Tivoli Directory Server Administration and Use for z/OS*
MVS programmer	• (Optionally) Creates VSAM object store and ICL data sets and indexes • (Optionally) sets up VSAM RLS • Starts the PKI Services daemon	• Basic MVS skills – Editing a data set – ISPF COPY command – MVS console START command • JCL knowledge to change job card • Basic browser and web skills	• *z/OS MVS System Commands*
RACF administrator	• Adds groups and user IDs • Sets up access control • Creates certificates • Sets up daemon security	• RACF administration • REXX skills (for working with IKYSETUP REXX exec) • RACF commands such as: – ADDGROUP – ADDSD – ADDUSER – RACDCERT – RDEFINE – PERMIT – SETROPTS • TSO skills	• *z/OS TSO/E REXX Reference* • *z/OS UNIX System Services Planning* • *z/OS Security Server RACF Security Administrator's Guide*

Table 5. Roles, tasks, and skills for setting up PKI Services (continued)

Role	Tasks	Required skills	Documented in:
UNIX programmer	• Copies files • (Optionally) customizes environment variables • (Optionally) customizes (non-**LDAP** sections of) `pkiserv.conf` configuration file • Sets up `/var/pkiserv` directory • Updates the **LDAP** section of the `pkiserv.conf` configuration file	• Basic UNIX commands, such as the **cp** (copy) command • Getting superuser authority	• *z/OS UNIX System Services Command Reference* • *z/OS UNIX System Services Planning*
Web server programmer	• Helps set up PKI Services – If you are implementing the PKI Services web application using REXX CGI execs: Updates the IBM HTTP Server - Powered by Apache configuration files and starts the IBM HTTP Server - Powered by Apache – If you are implementing the PKI Services web application using JavaServer pages (JSPs), updates the enterprise archive (EAR) file and deploys updated JSP files to a WebSphere Application Server. • Customizes the PKI Services web pages	• IBM HTTP Server - Powered by Apache customization skills • WebSphere Application Server customization and administration skills • Editing configuration files • Customizing the web pages	WebSphere Application Server Knowledge Center (www.ibm.com/support/knowledgecenter/SSEQTP)

Creating an implementation plan

Your implementation plan should include major subtasks, responsible parties, and a realistic estimate of time and effort required. The major tasks for implementing PKI Services are provided here as a basis for you to build your own plan.

Planning considerations for installing and configuring PKI Services

Table 6 on page 19 contains information that might affect how PKI Services is installed and configured. These concerns can help you to assess the size of your administrative staff, determine what required products might be needed, and what features of PKI Services might need to be used to meet your needs.

Table 6. Considerations before installing and configuring PKI Services

Considerations	**Affects on PKI Services configuration and operation**	**Topic**
Capability considerations: What types of certificates does this installation of PKI Services support? Does this installation provide customized certificate types or unique certificate types?	Impacts how the template files `pkiserv.tmpl` and `pkitmpl.xml` are modified when deciding what certificate types to support, and whether any customized types are supported.	See "Supported certificate types" on page 7 for the list of supported certificate types. See Chapter 11, "Customizing the end-user web application if you use REXX CGI execs," on page 135 for more information about the `pkiserv.tmpl` file.
Does this installation of PKI Services need to be able to generate key pairs on behalf of the certificate requester?	Deciding whether to configure PKI Services to generate key pairs determines whether an ICSF programmer is needed, and what tasks are needed to be performed. This also affects how the IKYSETUP script is customized.	See "Installing and configuring ICSF" on page 33.

Table 6. Considerations before installing and configuring PKI Services (continued)

Considerations	**Affects on PKI Services configuration and operation**	**Topic**
How do your PKI Services administrators and clients request and manage certificates? Does this installation provide a web browser interface to your clients?	Providing a browser interface requires a choice between using JavaServer Pages (JSPs) or REXX CGI execs to implement the web pages. This choice decides whether WebSphere Application Server or the IBM HTTP Server (IHS) is used to handle the web browser traffic. A web server programmer is needed to configure either server. Choosing a web browser interface might require updates to the `pkiserv.tmpl` for REXX CGI execs or `pkitmpl.xml` and JSPs installed by PKI Services. This is to support the specific features that this installation wants to provide to administrator and clients. This impacts how the template files `pkiserv.tmpl` and `pkitmpl.xml` are modified.	For assistance on customizing the IBM HTTP Server (IHS), see Chapter 7, “Updating IBM HTTP Server - Powered by Apache configuration and starting the server,” on page 101. Chapter 13, “Implementing the web application using JavaServer pages,” on page 237 contains information about customizing the default JSPs. See Chapter 11, “Customizing the end-user web application if you use REXX CGI execs,” on page 135 for more information about the `pkiserv.tmpl` file.
Does this installation allow certificate management through SCEP, EST, OCSP, or CMP protocols?	Supporting SCEP, EST, OCSP, or CMP protocols requires that the IBM HTTP Server (IHS) is enabled, regardless of the choice that is made for any web browser support. A web server programmer must configure the IBM HTTP Server.	For more information about using SCEP, EST, and OCSP, see Chapter 14, “Advanced customization,” on page 283. For more information about CMP, see Chapter 21, “Using the certificate management protocol (CMP) with PKI Services,” on page 453.

Table 6. Considerations before installing and configuring PKI Services (continued)

Considerations	Affects on PKI Services configuration and operation	Topic
Does this installation need to provide certificate revocation information. If so, does it provide this information through OCSP protocols or through certificate revocation lists (CRLs)?	To provided CRL information, modifications might be required to the PKI Services configuration file `pkiserv.conf`. To provided OCSP information, modifications might be required to the PKI Services template file `pkiserv.tmpl` or `pkitmpl.xml`. A z/OS UNIX System Services programmer might be needed if CRLs are stored to a z/OS UNIX System Services directory. New PKI Services installations should use this feature. Existing PKI Services installations should enable and upgrade to this feature. An LDAP programmer might be needed if CRLs are to be posted to an LDAP directory server. If certificate revocation information is made available through OCSP or CRL using http protocol, a web server programmer must configure the IBM HTTP Server - Powered by Apache.	See "Optionally updating the pkiserv.conf configuration file" on page 72 for details about the options for the `pkiserv.conf` file. Assistance for configuring the LDAP server for PKI Services is provided in Chapter 6, "Tailoring the LDAP configuration for PKI Services," on page 97. To store CRL information in a z/OS UNIX System Services directory, see "Enabling support for large CRLs" on page 297. Chapter 7, "Updating IBM HTTP Server - Powered by Apache configuration and starting the server," on page 101 contains descriptions of the IBM HTTP Server configuration features required by PKI Services.
Do you want to run in FIPS mode?	Ensure that the signing algorithm and the CA certificates fulfill requirements corresponding to the desired FIPS level specified in the `pkiserv.envars` file.	See Chapter 2 section, "Requirements for FIPS" on page 12 and Chapter 5 section, "(Optional) Steps for updating PKI Services environment variables" on page 72.

Table 6. Considerations before installing and configuring PKI Services (continued)

Considerations	**Affects on PKI Services configuration and operation**	**Topic**
Scaling considerations: What volume of certificate requests is this installation expected to handle? How many certificates is this installation expected to manage and track?	The expected volume of request traffic and managed certificates affects how many PKI Services administrators are needed to provide complete coverage. The number of PKI Services administrators affects how the sample IKYSETUP script is modified for this installation. This consideration also impacts the choice of backend storage for PKI Services: the default VSAM storage mechanism for typical installations; the optional Db2 storage mechanism for large-scale installations. If Db2 is used as the backend storage, Db2 must be installed before installing and configuring PKI Services, and the services of a Db2 database administrator is required. Also, the more certificates that are issued by a CA, the more certificates that can be potentially revoked. New PKI Services installations should choose to use the large CRL support in PKI Services; existing PKI Services installations should consider enabling and upgrading to the large CRL support. A z/OS UNIX System Services programmer is needed to enable and configure this feature. Large scale PKI Services installations that can generate a large volume of certificates or CRLs also impacts the planning tasks of the LDAP programmer, who must account for the storage that these certificates and CRLs can consume in LDAP.	Chapter 4, “Running IKYSETUP to perform RACF administration,” on page 39 contains information about how to tailor the IKYSETUP script for the needs of your installation. To set up VSAM as the backing storage, see “Creating the object store and ICL using VSAM data sets” on page 116 and the IKYSETUP instructions in Chapter 4, “Running IKYSETUP to perform RACF administration,” on page 39. To set up Db2 as the backing storage when Db2 is installed and configured, see “Creating the object store and ICL using Db2 tables” on page 122. To correctly modify IKYSETUP to configure PKI Services for Db2, see “Deciding the value of db2_repos and db2_subsys” on page 47. For instructions about enabling large CRL support and storing CRLs in a z/OS UNIX System Services directory, see “Enabling support for large CRLs” on page 297. For information about configuring LDAP to support PKI Services, see Chapter 6, “Tailoring the LDAP configuration for PKI Services,” on page 97.
Do you plan to suspend and resume certificates as a regular task in your normal operation?	If suspension and resumption of certificates are a regular occurrence, this increases the number of revoked certificates, and also factors in to the CRL support that is chosen for this installation.	For instructions on storing large CRLs, see “Enabling support for large CRLs” on page 297.

Table 6. Considerations before installing and configuring PKI Services (continued)

Considerations	Affects on PKI Services configuration and operation	Topic
Structural considerations: Is more than one CA hosted by a single z/OS image?	PKI Services can support multiple concurrently operating CAs within a single installation. Each CA operates within its own PKI Services CA domain.	Information about using and establishing multiple CA domains is in "Adding a new CA domain" on page 302.
Is this installation of PKI Services acting as a root certificate authority (CA) or an intermediate certificate authority?	By default, PKI Services is configured to be a single root certificate authority. PKI Services can be configured to be an intermediate CA using additional procedures.	The procedure for changing PKI Services to an intermediate CA are in "Establishing PKI Services as an intermediate CA" on page 486.
Procedural considerations: Do clients of PKI Services expect around-the-clock support? What vetting procedures are used to determine whether certificate requests should be approved or rejected? Do any of these procedures require the approval of multiple parties?	Administrative coverage and vetting procedures influence how many people are designated to be PKI Services administrators. The identities of PKI Services administrators are needed to correctly update the IKYSETUP script. If multiple parties are involved in vetting individual requests, the templates that are used for those requests in the `pkiserv.tmpl` and `pkitmpl.xml` files should be updated to enable multiple administrative approvals.	Chapter 4, "Running IKYSETUP to perform RACF administration," on page 39 contains information about how to tailor the IKYSETUP script for the needs of your installation. For more information about granting administrative access to users, see "Authorizing administrative functions" on page 497. See Chapter 11, "Customizing the end-user web application if you use REXX CGI execs," on page 135 for more information about the `pkiserv.tmpl` file. Multiple party approval is described in the ADMINNUM=*value*, where *value* is a numeric subsection of the template.

Table 6. Considerations before installing and configuring PKI Services (continued)

Considerations	**Affects on PKI Services configuration and operation**	**Topic**
What procedures indicate whether certificates should be suspended, revoked, or renewed? Do any of these procedures require the approval of multiple parties?	If certain requests can be automatically approved or rejected, or if certain certificates are allowed to be automatically renewed, the templates that are used for those requests in the `pkiserv.tmpl` and `pkitmpl.xml` files should be updated to enable these automated actions. If multiple parties are involved in vetting individual requests, the templates that are used for those requests should be updated to enable multiple administrative approvals.	See Chapter 11, "Customizing the end-user web application if you use REXX CGI execs," on page 135 for more information about the `pkiserv.tmpl` file. Also, included are instructions to set templates for automated approval or rejection. Multiple party approval is described in the ADMINNUM=*value*, where *value* is a numeric subsection of the template.
Do your procedures allow anyone that is considered to be a PKI Services administrator to perform any of the available administrative actions?	If there is a separation of duties between different PKI Services administrators (for example, if the administrators that approve browser certificates are not given authority to approve server certificates too), need to enable PKI Services granular authorization controls. The IKYSETUP script sets the `pkiserv.conf` file to enable these granular controls. The RACF programmer must create the necessary RACF resource profiles to control PKI Services administrator authority.	To set up PKI Services to use granular administrative controls, see "Deciding the value of AdminGranularControl" on page 48. The specific control for this option is described in Table 21 on page 74. The RACF classes and resources necessary to implement these controls are in "Using the PKISERV class to control access to administrative functions" on page 499.

Table 6. Considerations before installing and configuring PKI Services (continued)

Considerations	**Affects on PKI Services configuration and operation**	**Topic**
Administrative considerations: What is the best way to structure the group of PKI Services administrators? If multiple CAs are being hosted by a single z/OS image, are certain PKI Services administrators responsible only for specific CAs? Are certain PKI Services administrators authorized to perform only a subset of the administrative tasks? Are certain PKI Services administrators authorized to perform any administrative task?	Administrative coverage and vetting procedures influence how many people are designated to be PKI Services administrators. The identities of PKI Services administrators are needed to correctly update the IKYSETUP script. If any PKI Services administrators are being created only to monitor and audit the system, those administrators require READ access to the IRR.PKISERV.PKIADMIN and related RACF facility resource. PKI Services administrators that handle certificates and requests require UPDATE access to these RACF resources. The level of authority for each administrator affects how the IKYSETUP script is modified for this installation. If there is a separation of duties between different PKI Services administrators (for example, if the administrators that approve browser certificates are not given authority to approve server certificates too), need to enable PKI Services granular authorization controls. The IKYSETUP script sets the `pkiserv.conf` file to enable these granular controls. The RACF programmer must create the necessary RACF resource profiles to control PKI Services administrator authority.	Chapter 4, "Running IKYSETUP to perform RACF administration," on page 39 contains information about how to tailor the IKYSETUP script for the needs of your installation. For more information about granting administrative access to users, see "Authorizing administrative functions" on page 497. To set up PKI Services to use granular administrative controls, see "Deciding the value of AdminGranularControl" on page 48. The specific control for this option is in Table 21 on page 74. The RACF classes and resources necessary to implement these controls are in "Using the PKISERV class to control access to administrative functions" on page 499.

Table 6. Considerations before installing and configuring PKI Services (continued)

Considerations	Affects on PKI Services configuration and operation	Topic
How frequently does this installation handle SCEP-based requests for certificates?	How frequently your installation processes SCEP-based certificate requests are of concern for administrators of existing PKI Services installations that are upgrading to PKI Services Version 2 Release 3 or later. This question affects your decision on the format version to use for the Object Store and ICL, which is determined from the `DBVersion`, `ObjectSCEPTidDSN`, and `ICLSCEPTidDSN` keyword in the `pkiserv.conf` configuration file. If SCEP-based requests are rarely handled, you do not need to convert your existing Object Store and ICL from their existing format. You can continue to use the existing format by leaving `DBVersion`, `ObjectSCEPTidDSN`, and `ICLSCEPTidDSN` unset. For installations that frequently handle SCEP-based certificate requests, consider converting your existing Object Store and ICL to the new format version to allow for improved tracking of these requests. New installations of PKI Services should set up their installation to use the enhanced requester SCEP functionality that is provided in PKI Services Version 2 Release 3 and later.	For more information about SCEP-based requests for certificates, see "Creating the object store and ICL using VSAM data sets" on page 116 and "Creating the object store and ICL using Db2 tables" on page 122.

Task roadmap for implementing PKI Services

Table 7 on page 26 shows the subtasks and associated procedures for implementing PKI Services. These tasks comprise of the major part of your implementation plan.

Table 7. Task roadmap for implementing PKI Services

Subtask	Associated procedure (See ...)
Installing and configuring prerequisite products:	Chapter 3, "Installing and configuring prerequisite products," on page 29
• "IBM HTTP Server - Powered by Apache (optional)" on page 13	• "Steps for installing and configuring the IBM HTTP Server - Powered by Apache to work with PKI Services " on page 30
• "OCSF" on page 13	• "Installing and configuring OCSF" on page 35

Table 7. Task roadmap for implementing PKI Services (continued)

Subtask	Associated procedure (See ...)
• “LDAP directory server” on page 13	• “Steps for installing and configuring LDAP” on page 31
• “ICSF” on page 14	• “Installing and configuring ICSF” on page 33
• “sendmail (optional)” on page 14	• “Configuring sendmail (optional)” on page 34
• “Db2 (optional)” on page 14	• “Installing and configuring Db2” on page 34
Configuring your system for PKI Services:	Part 2, “Configuring your system for PKI Services,” on page 37
• RACF	• Chapter 4, “Running IKYSETUP to perform RACF administration,” on page 39
• z/OS UNIX	• Chapter 5, “Configuring the UNIX runtime environment,” on page 67
• LDAP configuration	• Chapter 6, “Tailoring the LDAP configuration for PKI Services,” on page 97
• HTTP server	• Chapter 7, “Updating IBM HTTP Server - Powered by Apache configuration and starting the server,” on page 101
• LDAP	• Chapter 8, “Tailoring the PKI Services configuration file for LDAP,” on page 107
• Set up the object store and issued certificate list (ICL)	• Chapter 9, “Creating the object store and ICL,” on page 115 • Chapter 10, “Starting and stopping PKI Services,” on page 129
Customizing PKI Services:	Part 3, “Customizing PKI Services,” on page 133
• Customizing end-user web pages	• Chapter 11, “Customizing the end-user web application if you use REXX CGI execs,” on page 135
• Customizing administration web pages	• Chapter 12, “Customizing the administration web pages if you use REXX CGI execs,” on page 233
• Advanced customizing	• Chapter 14, “Advanced customization,” on page 283
Testing PKI Services:	Part 4, “Using PKI Services,” on page 373
• Using end-user web pages	• Chapter 18, “Using the end-user web pages,” on page 375
• Using administration web pages	• Chapter 19, “Using the administration web pages,” on page 405

Table 7. Task roadmap for implementing PKI Services (continued)

Subtask	Associated procedure (See ...)
Administering PKI Services:	Part 5, “Administering security for PKI Services,” on page 479
• RACF	• Chapter 22, “RACF administration for PKI Services,” on page 481

Chapter 3. Installing and configuring prerequisite products

After the MVS programmer installs PKI Services using SMP/E (but before team members set up PKI Services, see Chapter 4, “Running IKYSETUP to perform RACF administration,” on page 39 through Chapter 9, “Creating the object store and ICL,” on page 115), your team needs to set up prerequisite products:

- An HTTP server - to handle requests through a web server
- WebSphere Application Server for z/OS (optional)- provides the web server if you implement the PKI Services web application using JavaServer pages (JSPs)
- LDAP - for posting certificates and CRLs
- ICSF - provides cryptographic functions that PKI Services will rely on, such as the storage of the CA's private key in hardware and the generation and storage of key pairs for PKI Services certificate requests.
- sendmail (optional) - for sending email notifications to certificate requestors and administrators
- OCSF (optional) - used by the PKI Trust Policy (PKITP)
- OCEP (optional) - used by the PKI Trust Policy (PKITP)
- Db2 (optional) - can be used to store the object store and issued certificate list (ICL) if VSAM data sets are not used

You need to install and configure the HTTP server and LDAP only if you are setting up prerequisite products for PKI Services for the first time.

Tasks to perform before setting up PKI Services

Before you can set up PKI Services, your team needs to set up prerequisite software products by completing the following tasks, if not already done:

1. “Installing and configuring the IBM HTTP Server - Powered by Apache” on page 29
2. “Installing and configuring WebSphere Application Server for z/OS” on page 31
3. “Installing and configuring LDAP” on page 31
4. “Installing and configuring ICSF” on page 33
5. “Configuring sendmail (optional)” on page 34
6. “Installing and configuring Db2” on page 34

This topic explains these tasks in more detail.

Installing and configuring the IBM HTTP Server - Powered by Apache

You need to perform this task only if you are setting up prerequisite products for PKI Services for the first time.

PKI Services requires that you have the IBM HTTP Server - Powered by Apache installed and configured for at least non-SSL page retrieval. Tasks of other team members, such as the RACF administrator and web server programmer (see Chapter 4, “Running IKYSETUP to perform RACF administration,” on page 39 and Chapter 7, “Updating IBM HTTP Server - Powered by Apache configuration and starting the server,” on page 101) assume that this is already done.

Steps for installing and configuring the IBM HTTP Server - Powered by Apache to work with PKI Services

Before you begin:

1. You need web server programming skills to complete this procedure.
2. You might need to refer to the documentation for the IBM HTTP Server - Powered by Apache which available in the WebSphere Application Server Knowledge Center (www.ibm.com/support/knowledgecenter/SSEQTP).

Procedure

Perform the following steps to install and configure the IBM HTTP Server - Powered by Apache to work with PKI Services:

1. Use the following table to decide what you need to do:

If ...	Then ...	Notes
The IBM HTTP Server - Powered by Apache is not installed and configured ...	Install and configure IBM HTTP Server - Powered by Apache by following the instructions in the installation section of IBM HTTP Server - Powered by Apache documentation..	**Guideline:** For PKI Services, when you install the IBM HTTP Server - Powered by Apache, do not use a password file.
The IBM HTTP Server - Powered by Apache is installed but not configured for SSL ...	Fill in the missing values in the table in the next step. (The RACF programmer needs information for setting up PKI Services; see Chapter 4, "Running IKYSETUP to perform RACF administration," on page 39.)	—
The IBM HTTP Server - Powered by Apache is installed and configured for SSL using a RACF key ring ...	Fill in the missing values in the table in the next step. (The RACF programmer needs information for setting up PKI Services; see Chapter 4, "Running IKYSETUP to perform RACF administration," on page 39.)	—
The IBM HTTP Server - Powered by Apache is installed and configured for SSL using gskkyman ...	Fill in the missing values in the table in the next step. (The RACF programmer needs information for setting up PKI Services; see Chapter 4, "Running IKYSETUP to perform RACF administration," on page 39. The RACF programmer also needs to add your CA certificate to an existing keyfile; see Appendix B, "Using a gskkyman key database for your certificate store," on page 689 for information about gskkyman steps.)	—

You can now perform the steps for the decision you have made.

____

2. Fill in the rightmost column of the following table with information from the configuration:

Table 8. IBM HTTP Server - Powered by Apache information you need to record

IBM HTTP Server - Powered by Apache information	Explanation	Value
IBM HTTP Server - Powered by Apache fully qualified domain name	A fully qualified domain name is the name of a host system. It includes a series of subnames (each of which is a domain name). For example, `ralvm7.vnet.ibm.com` is a fully qualified domain name that includes the domain names `ibm.com` and `vnet.ibm.com`. (The RACF administrator needs to know the fully qualified domain name when setting up PKI Services.)	
The full UNIX path name of your `httpd.conf` and `vhost`*`port_number`* configuration files	(The web server programmer needs to know the full UNIX path name when updating the `httpd.conf` and `vhost.conf` configuration files to support PKI Services.)	

Installing and configuring WebSphere Application Server for z/OS

You need to perform this task only if you are setting up prerequisite products for PKI Services for the first time, or if you are implementing the PKI Services web application using JavaServer pages (JSPs) for the first time.

If you implement the PKI Services web application using JavaServer pages (JSPs), you must install WebSphere Application Server with the supported version and configure an SSL configuration that uses client authentication. For more information, see Chapter 13, “Implementing the web application using JavaServer pages,” on page 237 and the WebSphere Application Server Knowledge Center (www.ibm.com/support/knowledgecenter/SSEQTP) for your release of WebSphere Application Server.

You do not need to install WebSphere Application Server if you implement the PKI Services web application using REXX CGI execs.

Installing and configuring LDAP

The LDAP programmer installs and configures LDAP for the TDBM Db2 backend and records entries that are needed later.

Steps for installing and configuring LDAP

You need to perform this task only if you are setting up prerequisite products for PKI Services for the first time.

Although it can be configured otherwise, typical PKI Services usage requires access to an LDAP directory server. Install the LDAP directory server separately from PKI Services. After the installation is complete, LDAP needs to be configured for PKI Services. The directory stores issued certificates and certification revocation lists. The z/OS LDAP server provided by IBM Tivoli Directory Server for z/OS is preferred but not required. The remainder of this topic assumes you are using the IBM Tivoli Directory Server for z/OS LDAP server.

Note: The default name of the LDAP server configuration file is `ds.conf`.

You can use a non-z/OS LDAP server if it can support the object classes and attributes that PKI Services uses. For information about using a non-z/OS LDAP server, see Appendix A, “LDAP directory server requirements,” on page 685.

Before you begin

1. You need LDAP programming skills to complete this procedure.
2. For more information, see *z/OS IBM Tivoli Directory Server Administration and Use for z/OS*.

Procedure

Perform the following steps to install and configure LDAP to work with PKI Services:

1. Use the following table to decide what you need to do:

If ...	Then...	Notes
You do not have LDAP installed and configured ...	Follow the instructions in the Administration section of *z/OS IBM Tivoli Directory Server Administration and Use for z/OS*.	
You have LDAP installed and configured but not for the TDBM or LDBM backend ...	You need to migrate to the TDBM or LDBM backend. See *z/OS IBM Tivoli Directory Server Administration and Use for z/OS* for details about how to do this.	--
You have LDAP installed and configured for the TDBM or LDBM backend ...	Go to the next step.	--

You can now perform the steps for the decision you have made.

____

2. Record the entries and values from the LDAP configuration step in the following table. (Your team needs this information when setting up PKI Services.)

Table 9. LDAP information you need to record

LDAP information	Explanation	Value
Distinguished name	This is the distinguished name to use for LDAP binding. A distinguished name is the unique name of a data entry that identifies its position in the hierarchical structure of the directory. A distinguished name consists of the relative distinguished name (RDN) concatenated with the names of its ancestor entries. For example, an entry for Tim Jones could have an RDN of `CN=Tim Jones` and a DN of: `CN=Tim Jones,O=IBM,C=US` Any RDN type supported by the LDAP server can be used. The distinguished name can be a RACF-style distinguished name. For information about RACF-style distinguished names, see *z/OS IBM Tivoli Directory Server Administration and Use for z/OS*. For example, an entry for RACF user ID `timjones` is: `RACFID=timjones,PROFILETYPE=user,O=racfdb,C=us`	
Distinguished name password	This is the password that is defined for the distinguished name, for use by PKI to bind to the LDAP server. RACF passwords can be case-sensitive, so make sure that the password specified for a RACF-style distinguished name in the pkiserv.conf file or in the LDAPBIND profile matches the RACF password exactly.	

Table 9. LDAP information you need to record (continued)

LDAP information	Explanation	Value
LDAP fully qualified domain name and port	This is the domain name on which the LDAP server is listening. For example, for `ldap.widgets.com:389`, the fully qualified domain name is `ldap.widgets.com` and the port is 389. See Table 8 on page 31 for a definition of fully qualified domain name.	
Suffix	A suffix in LDAP is the upper-level name of the subtree. For example, for the following distinguished name: `OU=your-CA's-friendly-name,O=your-organization, C=your-country-abbreviation` the suffix could be either "O=*your company*,C=*your-country-abbreviation*" or "C=*your-country-abbreviation*" The suffix value is specified after the suffix keyword in the LDAP server configuration file: `suffix "O=your-company,C=your-country-abbreviation"` **Note:** If you have more than one suffix, record the suffix you intend to use as the root for storing the PKI Services CA certificate.	

____

3. The topics that follow require the LDAP server to be running. Follow the instructions in the topic about running the LDAP server in *z/OS IBM Tivoli Directory Server Administration and Use for z/OS*.

____

Installing and configuring ICSF

PKI Services relies on ICSF to perform general cryptographic functions and provides key storage and generation as follows:

- The PKI Services CA signing key can be securely generated and stored in the ICSF's Public Key Data Set (PKDS) or Token Key Data Set (TKDS) if directed to do so. For this to be successful, the ICSF programmer must configure the ICSF CCA cryptographic coprocessor and the PKDS or configure ICSF PKCS#11 cryptographic coprocessor and the TKDS. (The RACF administrator uses the IKYSETUP REXX exec to set up any RACF profiles that are needed to control access to ICSF services and keys. For more information, see Chapter 4, "Running IKYSETUP to perform RACF administration," on page 39.)
- PKI Services uses ICSF's PKCS#11 function to generate key pair for the regular request and the Certificate Management Protocol(CMP) request. For the regular request, the key pair is stored in the Token Key Data Set (TKDS). Furthermore, if secure key is needed in the TKDS, the PKCS#11 cryptoprocessor is required. For the CMP request, TKDS is not required as the CMP CGI does not store keys.

Note: A CAA Cryptographic Coprocessor is needed if you want your PKI Services CA key to be protected by the master key in the Public Key Data Set (PKDS). A PKCS#11 Cryptographic Coprocessor is needed if you want your PKI Services CA key or the key generated by PKI Services to be protected by the master key in the Token Key Data Set (TKDS).

Before you begin

- You need ICSF programming skills to complete this procedure.
- You might need to refer to the following documents:
  - *z/OS Cryptographic Services ICSF Administrator's Guide*

This document provides information about managing cryptographic keys, setting up and maintaining the PKDS, controlling who can use cryptographic keys and services, and general information about ICSF and cryptographic keys.

– *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*

  This document describes the ICSF support for PKCS #11, and provides information about setting up the TKDS.

Procedure

If ICSF is not already installed and configured for PKA, do this by following the instructions in *z/OS Cryptographic Services ICSF Administrator's Guide*. If you want PKI Services to generate key pairs for certificate requests, set up the TKDS by following the instructions in *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*.

Configuring sendmail (optional)

The UNIX programmer needs to configure sendmail if your installation plans to use any of the following functions:

- Sending email notifications to users whose certificate request is rejected, ready for retrieval, or about to expire
- Sending email notifications to administrators who have requests pending
- Automatic certificate renewal
- Requesting certificates for which the key pair (public key and private key) is generated by PKI Services

Note: Before you begin, you need the following document:

- *z/OS Communications Server: IP Configuration Guide*

Follow the instructions in *z/OS Communications Server: IP Configuration Guide* for configuring the CSSMTP application sendmail. In general, you need to perform the following steps:

1. Configure and start CSSMTP.
2. Configure JES spool data set for CSSMTP.

Notes: Since the PKI Services daemon invokes the `sendmail` command to send email notifications, the PATH environment variable in the `pkiserv.envars` file must be updated to include the directory where the `sendmail` command is located. By default, the `sendmail` command is located in the `/bin` directory.

Once the CSSMTP application has been configured, perform the following steps to test your `sendmail` configuration:

1. From the UNIX command line, create a mail file named `mail.txt` with the following information. Replace `target-email@address.com` with the email address you want to send the email to, and replace the `source-email@address.com` with the email address of the sender of the email.

```
To:target-email@address.com
From:source-email@address.com
Subject:This is a test
```

2. Enter the command:

```
/bin/sendmail -t <mail.txt
```

Installing and configuring Db2

You need to perform this task only if you are implementing the PKI Services object store and issued certificate list (ICL) using Db2 tables for the first time.

For information about installing and configuring Db2, see DB2 for z/OS in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/SSEPEK/db2z_prodhome.html) for your release of Db2.

You do not need to install Db2 if you implement the PKI Services object store and issued certificate list (ICL) using VSAM data sets.

For information about implementing the PKI Services object store and ICL, see Chapter 9, “Creating the object store and ICL,” on page 115.

Tasks to perform before configuring PKITP

If you plan to use the PKI Services Trust Policy (PKITP), your team first needs to set up prerequisite software products by completing the following tasks, if not already done:

1. “Installing and configuring OCSF” on page 35
2. “Installing and configuring OCEP” on page 35

Installing and configuring OCSF

PKI Services Trust Policy (PKITP) requires OCSF to be installed and configured. If OCSF is not already installed and configured, follow the instructions for doing so in *z/OS Open Cryptographic Services Facility Application Programming*.

Before you begin

1. Although the base feature of z/OS includes OCSF, if you are in the United States or Canada, make sure that you ordered and installed the additional OCSF Security Level 3 feature. (There is no charge for this feature.)
2. You need OCSF programming skills to complete this procedure.

Installing and configuring OCEP

To install and configure OCEP, follow the instructions in *z/OS Integrated Security Services Open Cryptographic Enhanced Plug-ins Application Programming*.

You need to perform this task only if you are setting up prerequisite products for PKITP for the first time.

Part 2. Configuring your system for PKI Services

After the MVS programmer installs PKI Services into the file system directory, your team needs to perform additional tasks to configure PKI Services, including the following tasks:

- Chapter 4, “Running IKYSETUP to perform RACF administration,” on page 39 describes how the RACF administrator updates and runs IKYSETUP, a REXX exec to perform RACF administration tasks, such as setting up the daemon user ID and giving accesses.
- Chapter 5, “Configuring the UNIX runtime environment,” on page 67 explains:
  - Copying files, such as the PKI Services configuration file
  - Updating environment variables
  - Updating the PKI Services configuration file
  - Setting up the `/var/pkiserv` file system directory.
- Chapter 6, “Tailoring the LDAP configuration for PKI Services,” on page 97 explains how to update your LDAP configuration (performed earlier, see “Installing and configuring LDAP” on page 31) for PKI Services.
- Chapter 7, “Updating IBM HTTP Server - Powered by Apache configuration and starting the server,” on page 101 describes updating the IBM HTTP Server configuration files and starting the IBM HTTP Server.
- Chapter 8, “Tailoring the PKI Services configuration file for LDAP,” on page 107 explains how to update the **LDAP** section of the PKI Services configuration file.
- Chapter 9, “Creating the object store and ICL,” on page 115 explains how to create the VSAM data sets or Db2 objects for the PKI Services object store and issued certificate list (ICL).
- Chapter 10, “Starting and stopping PKI Services,” on page 129 explains how to start and stop the PKI Services daemon.

Chapter 4. Running IKYSETUP to perform RACF administration

You need to perform this task if you are configuring PKI Services for the first time or adding a new CA domain.

PKI Services provides SYS1.SAMPLIB(IKYSETUP), a REXX exec, to perform RACF administration tasks for setting up PKI Services. The RACF administrator updates and runs this REXX exec, which issues RACF commands to perform the following tasks:

- Adding groups and user IDs
  - Setting up the PKI Services administration group
  - Creating the PKI Services daemon user ID
  - Giving appropriate access to the RACF group
  - Creating the surrogate user ID and giving the surrogate user ID authority to generate certificates

    A surrogate user ID is the identity that is assigned to client processes when they are requesting certificate services. A surrogate user ID is required for external clients. **Guideline:** For simplicity, use surrogate user IDs for internal clients also, rather than allowing them to access PKI Services under their own identities.
  - Associating the PKI Services daemon user ID with the PKI Services started procedure.
- Setting up access control to protect end-user and administrative functions of PKI Services:
  - Authorizing the PKI Services daemon user ID for CA functions
  - Authorizing the PKI Services daemon user ID to access the Resource Recovery Services access facility (RRSAF), if you use Db2 as the repository for the object store and ICL
  - Giving administrators access to VSAM data sets, if you use VSAM as the repository for the object store and ICL
  - Authorizing PKI Services for ICSF resources.
  - Optionally defining granular administrative controls
- Creating certificate authority (CA), registration authority (RA), and SSL certificates:
  - Creating a CA certificate and private key
  - Backing them up to a password-protected MVS data set
  - Optionally migrating the private key to ICSF
  - Optionally creating an RA certificate and private key for Simple Certificate Enrollment Protocol (SCEP)
  - Creating a SAF key ring and associating it with the certificate
  - Exporting the CA certificate to an MVS data set and file system file
  - Generating a server certificate signed by the new CA
  - Creating a key ring for the web server
  - Associating the web server and any trusted CA certificates to the key ring.
- Setting up the IBM HTTP Server for surrogate operation.
- Allowing PKI Services to generate key pairs for certificate requests

Overview of IKYSETUP

IKYSETUP consists of several parts:

- A configurable section. This section assigns values to variables.

- A section that issues RACF commands to perform RACF administration tasks. (See "Actions IKYSETUP performs by issuing RACF commands" on page 619 for details about the actions that various sections of code perform.)
- A section that writes information (such as the name of the PKI Services administration group) to the log data set. The log itself consists of two parts: commands that are issued and other information. (See "Sample IKYSETUP log data set" on page 63.)

  Note: By default, IKYSETUP creates the log. You can disable recording information to the log by changing the value of one of the variables in IKYSETUP (`log_dsn`) to null.

The configurable section contains three parts:

- Values that you must change (by making them specific to your company, such as your company's name)
- Values that you might change depending on how you want PKI Services set up
- Values that you can optionally change (these defaults are acceptable without change, but you might want to change them to make them more specific to your company, for example the name of the PKI Services administration group, which by default is PKIGRP)

The following table illustrates the structure and divisions of IKYSETUP:

Table 10. IKYSETUP structure and divisions

Structure and divisions of IKYSETUP
Configurable section, assigns values to variables • Values that you must change to customize (See Table 11 on page 41.) • Values that you might change that are related to set up (See Table 18 on page 50.) • Values that you can optionally change (See Table 19 on page 55.)
Issues RACF commands
Records information in the log data set

Before you begin

Important: Update and run IKYSETUP *only* if you have not done so previously for an earlier release (or if you are changing the value of one or more variables).

You need to collect the following documents:

- *z/OS Security Server RACF Command Language Reference*
- *z/OS Security Server RACF Security Administrator's Guide*
- *z/OS TSO/E REXX Reference*

The RACF administrator needs to decide the values of variables in IKYSETUP and to record these values for future reference. Review and update as necessary the following three variables tables. There are three tables because there are three categories of variables:

- Variables whose values you are *required* to change, such as ones containing your company name
- Variables whose values you might want to change, depending based on how you are setting up PKI Services
- Variables whose values you can optionally change.

There is some overlap between the three types of variables, for example, if you are already using the RACF sample web application, PKISERV.

Guideline: If you are running IKYSETUP for the first time, at a minimum, you need to complete the following tasks:

- Fill in Table 11 on page 41
- Fill in Table 15 on page 47
- Fill in the rows of Table 18 on page 50 concerning z/OS UNIX level security:
  - `unix_sec`
  - `bpx_userid.` and `pgmcntl_dsn.` (if z/OS level security is already set up)
- Review the default values in *all* the tables.

Several values that are described in this topic, particularly for variables in Table 11 on page 41 and Table 19 on page 55, can be qualified with a *ca_domain* value based on whether you implement multiple CA domains. For information about implementing multiple CA domains, see "Adding a new CA domain" on page 302.

Variables whose values must change

Fill in the blank lines in the rightmost column with your company's information (and cross out the defaults in these cells).

Table 11. IKYSETUP variables whose values must change

Variable name	**Description**	**Referenced elsewhere**	**Default value and your company's information**

`ca_dn`	The CA's distinguished name. (For a definition of distinguished name, see Table 9 on page 32.)
	
	If you already have your CA certificate and private key set up in RACF, set `ca_dn=""`, set `ca_label` (in the following row) to the value of your CA's label, and update `ca_expires` (in Table 19 on page 55) to reflect the expiration date of your CA certificate.
	
	If you do not already have your CA certificate and private key set up in RACF, cross out the default in the rightmost cell of this row and record the information for your company-specific information for distinguished name on the blank line.
		The suffix of the PKI Services CA's distinguished name must match the LDAP suffix. (The LDAP suffix is in the LDAP server configuration file. See Table 9 on page 32 for a definition of suffix.)
		
		Note: However, do not specify a C('*value*') if it is not present in your LDAP suffix.
			When you also set *ca_domain*: `OU('`*ca_domain* `Human Resources Certificate Authority')`
			
			When you do not set *ca_domain*: `OU('Human Resources Certificate Authority')`
			
			`O('Your Company')`
			
			`C('Your Country 2 Letter Abbreviation')`
			
			______________________________

Table 11. IKYSETUP variables whose values must change (continued)

Variable name	**Description**	**Referenced elsewhere**	**Default value and your company's information**
`ca_label`	The CA certificate label. If you already have your CA certificate and private key set up in RACF (and your CA certificate's label differs from the default), you need to set `ca_label` to your CA certificate's label.	No	When you also set *ca_domain*: ***ca_domain*** `Local PKI CA` When you do not set *ca_domain*: `Local PKI CA` (Replace the default if you already have your CA certificate and private key set up in RACF.) ________________________________
`daemon_uid`	The z/OS UNIX user identifier (UID) associated with the PKI Services daemon user ID.	No	**554** ________________________________
`pki_gid`	The z/OS UNIX group identifier (GID) for the PKI Services administration group.	No	**655** ________________________________
`pkigroup_mem`	Members of the PKI administration group are responsible for administering PKI Services functions. **Guideline:** Assign PKI administration duties to only highly trusted individuals. `pkigroup_mem.` is a list in which `pkigroup_mem.0` is the number of members in the list and the rest of the entries are their user IDs. You must change the `pkigroup_mem.0` to at least 1, and change `pkigroup_mem.1` through `pkigroup_mem.`*n* to the member user IDs.	No	`0` (default for `pkigroup_mem.0`, the number of member user IDs) ________________________________ **Note:** You must change the default to at least 1. (Record the member IDs:) ________________________________ ________________________________ ________________________________ ________________________________ ________________________________

Table 11. IKYSETUP variables whose values must change (continued)

Variable name	Description	Referenced elsewhere	Default value and your company's information

`ra_dn`	The RA's distinguished name for use with Simple Certificate Enrollment Protocol (SCEP). (For a definition of distinguished name, see Table 9 on page 32.)
	
	This name should be similar but not identical to your CA's distinguished name. If you do not want to have PKI Services operate with a separate RA certificate, set `ra_dn=""`.
		No	`CN('Registration Authority')`
			
			`OU('Human Resources Certificate Authority')`
			
			`O('Your Company')`
			
			`C('Your Country 2 Letter Abbreviation')`
			
			____________________________

`ra_label`	The certificate label of your RA certificate in RACF.	No	When you also set ca_domain:*ca_domain*`Local PKI RA` When you do not set ca_domain: `Local PKI RA` ____________________________
`surrog_uid`	The UID associated with the surrogate user ID.	No	**555** ____________________________

Table 11. IKYSETUP variables whose values must change (continued)

Variable name	Description	Referenced elsewhere	Default value and your company's information

`web_dn`	Your web server's distinguished name. (For a definition of distinguished name, see Table 9 on page 32.)
	Notes:
	1. The RACF administrator copies the fully qualified domain name from an earlier table: Table 8 on page 31.
	2. If you already have your web server configured for SSL:
	• Set `web_dn=""`
	• Update the `web_ring` row
	(You need to connect your PKI Services CA certificate to your key ring. See the `web_ring` row for directions.)
		The value of the web server's common name (CN), which is your server's symbol IP address. For example, www.*YourCompany*.com must match your web server's fully qualified domain name.	`CN('www.YourCompany.com')`
			`O('Your Company')`
			`L('Your City')`
			`SP('Your Full State or Province Name')`
			`C('Your Country 2 Letter Abbreviation')`
			________________________________

`web_ring`	The name of the web server's SAF key ring.
	If your web server is configured for SSL and you are using a RACF key ring, set `web_ring` to the value of the RACF key ring. If your web server is configured for SSL and you are using gskkyman, set `web_ring=" "` and see Appendix B, "Using a gskkyman key database for your certificate store," on page 689 for additional directions.
		`vhost443.conf`
		Host file for SSL requests with server authentication
		`vhost1443.conf`
		Host file for SSL requests with client authentication
			`SSLring`
			________________________________

Variables whose values might change depending on setup

To help in completing the next table of variables (see Table 18 on page 50) fill out the decision tables in "Deciding the value of key_backup" on page 45, "Deciding the value of key_type" on page 45, "Deciding the value of restrict_surrog" on page 46, "Deciding the value of unix_sec" on page 47, and "Deciding the value of db2_repos and db2_subsys" on page 47.

Deciding the value of key_backup

Use the following decision table to determine the value of `key_backup` in Table 18 on page 50. The `key_backup` variable determines whether the PKI Services CA certificate and private key should be backed up to an encrypted data set.

Table 12. Decision table for key_backup

If ...	Then ...	Notes
You want to back up your CA's certificate and private key to a passphrase encrypted data set ...	Do not change the default `key_backup=1`	When you use IKYSETUP, you need to enter a passphrase whose display is not inhibited, it appears on the screen in the clear. You cannot back up the key stored in the PKDS or TKDS (`key_type=2,6,7,8,9,10`) to a data set.
You do *not* want to back up your CA's certificate and private key to a data set ...	Set `key_backup=0`	—

Deciding the value of key_type

Use the following decision table to determine the value of `key_type` in Table 18 on page 50. The `key_type` variable determines the key type (RSA, DSA, NIST ECC or BP ECC) and the location (RACF, PKDS or TKDS) of the private key.

By default, IKYSETUP generates RSA key pair using software cryptography.

Guideline: Do not change the default the first time you run IKYSETUP but change it before going into a production environment. (For information about installing and configuring ICSF, see "Installing and configuring ICSF" on page 33.)

Table 13. Decision table for key_type

If ...	Then	Notes
You want to generate the RSA key pair using software cryptography and store the private key in RACF ...	Do not change the default `key_type=0`	—
You want to generate the RSA key pair using the ICSF CCA cryptographic coprocessor and store the private key in Public Key Data Set (PKDS) ... **Note:** This key type is not FIPS compliant.	Set `key_type=2`	PKI Services does not automatically back up the private key when you select the 2 value. Review and possibly change the following additional variables in Table 18 on page 50: • `csfkeys_profile` • `csfserv_profile` • `csfusers_grp`
You want to generate the DSA key pair using software cryptography and store the private key in RACF ...	Set `key_type=3`	
You want to generate the NIST ECC key pair using software cryptography and store the private key in RACF ...	Set `key_type=4`	

Table 13. Decision table for key_type (continued)

If ...	Then	Notes
You want to generate the Brainpool (BP) ECC key pair using software cryptography and store the private key in RACF ... **Note:** This key type is not FIPS compliant.	Set `key_type=5`	
You want to generate the NIST ECC key pair using the ICSF CCA cryptographic coprocessor and store the private key in Public Key Data Set (PKDS) ... **Note:** This key type is not FIPS compliant.	Set `key_type=6`	Review and possibly change the following additional variables in Table 18 on page 50: • `csfkeys_profile` • `csfserv_profile` • `csfusers_grp`
You want to generate the Brainpool (BP) ECC key pair using the ICSF CCA cryptographic coprocessor and store the private key in Public Key Data Set (PKDS) ... **Note:** This key type is not FIPS compliant.	Set `key_type=7`	Review and possibly change the following additional variables in Table 18 on page 50: • `csfkeys_profile` • `csfserv_profile` • `csfusers_grp`
You want to generate the RSA key pair using the ICSF PKCS#11 cryptographic coprocessor and store the private key in Token Key Data Set (TKDS)	Set `key_type=8`	The key is stored in the token *daemon*.CATOKEN. For example, PKISRVD.CATOKEN.
You want to generate the NIST ECC key pair using the ICSF PKCS#11 cryptographic coprocessor and store the private key in Token Key Data Set (TKDS)	Set `key_type=9`	The key is stored in the token *daemon*.CATOKEN. For example, PKISRVD.CATOKEN.
You want to generate the Brainpool (BP) ECC key pair using the ICSF PKCS#11 cryptographic coprocessor and store the private key in Token Key Data Set (TKDS) **Note:** This key type is not FIPS compliant.	Set `key_type=10`	The key is stored in the token *daemon*.CATOKEN. For example, PKISRVD.CATOKEN.

Deciding the value of restrict_surrog

Use the following decision table to determine the value of `restrict_surrog` in Table 18 on page 50. The `restrict_surrog` variable determines if the RESTRICTED attribute is assigned to the surrogate user ID. The RESTRICTED attribute limits the resources available to this user ID.

By default, IKYSETUP does not assign the RESTRICTED attribute to the surrogate user ID. **Guideline:** Do not change the default the first time you run IKYSETUP but change it before going into a production environment. For more information, see the topic about defining groups and users in *z/OS Security Server RACF Security Administrator's Guide*.

Table 14. Decision table for restrict_surrog

If ...	Then ...
You want to assign the RESTRICTED attribute to the surrogate user ID ...	Set `restrict_surrog=1`
You do *not* want to assign the RESTRICTED attribute to the surrogate user ID ...	Do not change the default `restrict_surrog=0`

Deciding the value of unix_sec

Use the following decision table to determine the value of `unix_sec` in Table 18 on page 50. The `unix_sec` variable determines whether you want to use z/OS UNIX security, which is a higher level of security. z/OS UNIX provides two levels of security:

UNIX level security
This is a less stringent level of security than z/OS UNIX level security. It is for installations where system programmers have been granted superuser authority. Programs that run with superuser authority have daemon level authority and can issue MVS identity-changing services without entering a `_passwd()` for the target user ID. With this level of security, the BPX.DAEMON profile in the FACILITY class is not defined.

z/OS UNIX level security
This is a higher level of security than z/OS UNIX level security. It lets your system exercise more control over superusers. With this level of security, the BPX.DAEMON profile in the FACILITY class is defined.

Table 15. Decision table for unix_sec

If ...	Then ...	Notes
You already have z/OS UNIX security set up ...	Set `unix_sec=1`	—
You do not have z/OS UNIX security set up and you do not want to set it up ...	Do not change the default of `unix_sec=0`	—
You do not have z/OS UNIX security set up and you want to set it up for the first time ...	Set `unix_sec=2`	1. For information about additional manual configuration, see the section about establishing z/OS UNIX security in *z/OS UNIX System Services Planning*. 2. If you are setting `unix_sec=2`, you must update the following variables: • `bpx_userid.` • `pgmcntl_dsn.`

Deciding the value of db2_repos and db2_subsys

Use the following decision table to determine the value of `db2_repos` and `db2_subsys` in Table 18 on page 50. You have the option of using either VSAM data sets or Db2 tables as the repositories for the PKI Services object store and ICL. For information about these options, see Chapter 9, "Creating the object store and ICL," on page 115. If you use Db2, the PKI Services daemon needs authorization to the Db2 Resource Recovery Services Access Facility (RRSAF). If you use VSAM, the PKI Services daemon and administrator need authorization to the VSAM data sets.

Table 16. Decision table for db2_repos and db2_subsys

If ...	Then ...	Notes
You plan to use VSAM ...	Do not change the default of `db2_repos=0`. Do not change `db2_subsys`; it is ignored.	The value of `db2_repos` should be consistent with the value of the `DBType` parameter in the `pkiserv.conf` configuration file.
You plan to use Db2 ...	Set `db2_repos=1`. Set `db2_subsys` to the name of the local Db2 subsystem or the group attachment name for the object store and ICL.	The value of `db2_repos` should be consistent with the value of the `DBType` parameter in the `pkiserv.conf` configuration file. The value of `db2_subsys` should be the same as the value of the `DBSubsystem` parameter in the `pkiserv.conf` configuration file.

Deciding the value of AdminGranularControl

Use the following decision table to determine the value of `AdminGranularControl` in Table 18 on page 50. The `AdminGranularControl` variable determines whether the IKYSETUP exec creates profiles to control the additional administrative function access controls described in "Using the PKISERV class to control access to administrative functions" on page 499.

Note: The `AdminGranularControl` parameter in `pkiserv.conf` determines whether the additional administrative function access controls described in "Using the PKISERV class to control access to administrative functions" on page 499 are enabled.

Table 17. Decision table for `AdminGranularControl`

If ...	Then ...	Notes
You do not want to restrict certain administrative functions to certain PKI Services administrators and the value of the `AdminGranularControl` variable in `pkiserv.conf` is F ...	Do not change the default of `AdminGranularControl = 0`.	

Table 17. Decision table for `AdminGranularControl` (continued)

If ...	Then ...	Notes
You want to restrict certain administrative functions to certain PKI Services administrators and the value of the `AdminGranularControl` variable in `pkiserv.conf` is T ...	Set `AdminGranularControl = 1.`	You must also modify the `pkigroup1_mem` and possibly `pkigroup2_mem` array variables to specify the number of PKI Services administrators that are to be given access to administrative functions and their user IDs. Depending on the controls that you want to set, the number of PKI Services administrators, and the templates configured for the system, you might also need to add `template`*x*, `pkigroup`*x*, `pki_gid`*x*, `pkigroup`*x*`_mem`, and `actions`*x* variables to the IKYSETUP exec. For more information about granular control of administrator functions, see “Using the PKISERV class to control access to administrative functions” on page 499.

Table of IKYSETUP variables that you might want to change

Update Table 18 on page 50 based on your answers in the decision tables in “Deciding the value of key_backup” on page 45, “Deciding the value of key_type” on page 45, “Deciding the value of restrict_surrog” on page 46, “Deciding the value of unix_sec” on page 47, and “Deciding the value of db2_repos and db2_subsys” on page 47. If you have decided to change any of the defaults in the rightmost column, cross out the defaults and enter your company's information:

Table 18. IKYSETUP variables you might want to change depending on setup

Variable name	**Description**	**Referenced elsewhere**	**Default value or your company's information**
`AdminGranularControl`	Specifies whether IKYSETUP is to set up profiles in the PKISERV class to control authorization to administration functions. **0** Do not set up the profiles **1** Set up the profiles If set to 1, other variable names must be set. See "Deciding the value of AdminGranularControl" on page 48.	The `AdminGranularControl` keyword in `pkiserv.conf`	`0`
`bpx_userid`	A list of user IDs with daemon and server authority. The `bpx_userid.0` is the number of items in the list and the rest of the entries are the z/OS UNIX user IDs. (This is non-applicable if `unix_sec` is not set to 2.)	No	1 (default for number of items) `OMVSKERN`

Table 18. IKYSETUP variables you might want to change depending on setup (continued)

Variable name	Description	Referenced elsewhere	Default value or your company's information
`ca_keysize`	The size in bits of the certificate-authority's private key. **RSA key** `key_type=0` or `key_type=2`, the acceptable range is 512 to 4096; `key_type=8`, the acceptable range is 1024 to 4096. **DSA key** `key_type=3`, the acceptable range is 512 to 2048. **NIST ECC key** `key_type=4`, `key_type=6` or `key_type=9`, the acceptable values are 192, 224, 256, 384 and 521. **BP ECC key** `key_type=5`, `key_type=7` or `key_type=10`, the acceptable values are 160, 192, 224, 256, 320, 384 and 512.	No	`2048`
`cryptoz_grp`	The name of a RACF group for users who are authorized to use PKCS #11 tokens. Applies only if you set `key_gen = 1`.	No	
`csfkeys_profile`	A profile to protect the PKI Services key in ICSF. (This variable is non-applicable if `key_type` has the value 0, 3, 4, or 5.) If you do not want IKYSETUP to create the profile, set `csfkeys_profile=""`. **Note:** When RACF stores the private key in the PKDS, it generates the label as: `'IRR.DIGTCERT.CERTIFAUTH.` *`unique-time-stamp`*`'`	No	`IRR.DIGTCERT.CERTIFAUTH.*`

Table 18. IKYSETUP variables you might want to change depending on setup (continued)

Variable name	Description	Referenced elsewhere	Default value or your company's information
csfserv_profile	A profile to protect ICSF services. (This is non-applicable if key_type=0 or key_type=3 and key_gen=0.)	No	CSF*
csfusers_grp	A group of authorized ICSF service users. (This is non-applicable if csfkeys_profile="" and csfserv_profile="" .)	No	
db2_repos	Specifies whether the repository for the PKI Services object store and ICL is Db2 or VSAM. **0** VSAM **1** Db2	pkiserv.conf – DBType parameter	0 (VSAM)
db2_subsys	Specifies the name of the local Db2 subsystem or the group attachment name for the object store and ICL repositories, if db2_repos is 1. This is a name of 1 - 4 alphanumeric characters.	pkiserv.conf – DBSubsystem parameter	DSN9
key_backup	Specifies whether the PKI Services CA certificate and private key should be backed up to a data set. The value can be: **1** Yes - default **0** No **Note:** This value is ignored when the value of key_type is 2, 6, 7, 8, 9 or 10. When you use IKYSETUP with key_backup=1, you need to enter a passphrase whose display is not inhibited - it appears on the screen in the clear.	No	1 (yes)

Table 18. IKYSETUP variables you might want to change depending on setup (continued)

Variable name	Description	Referenced elsewhere	Default value or your company's information
`key_gen`	Specifies whether you want PKI Services to generate key pairs (public key and private key) for certificate requests, if asked to do so. Set to 1 to allow PKI Services to generate key pairs for certificate requests.	No	`0`
`key_type`	Refer to the Table 13 on page 45 for information.	No	`0`
`pgmcntl_dsn`	A list in which `pgmcntl_dsn.0` is the number of items in the list and the rest of the entries are a list of load libraries to be program controlled. **Rule:** If you set `unix_sec=2`, you must update the list of data sets.	No	8 (default for number of items) • 'CEE.SCEERUN' • 'CBC.SCLBDLL' • 'SYS1.SIEALNKE' • 'SYS1.CSSLIB' • 'TCPIP.SEZALOAD' • 'SYS1.LINKLIB' • 'CSF.SCSFMOD0' • 'CSF.SCSFMOD1'
`pkigroup1_mem` `pkigroup2_mem`	Lists of users who are connected to PKI Services administrative groups. `pkigroup1_mem.0` is the number of members to connect to `pkigroup1`, and `pkigroup1_mem.`*n* specifies the users to connect. These variables are ignored if `AdminGranularControl = 0`.	No	`""`

Table 18. IKYSETUP variables you might want to change depending on setup (continued)

Variable name	Description	Referenced elsewhere	Default value or your company's information
`restrict_surrog`	Specifies whether the surrogate user ID should be marked restricted. The value can be: **0** No - default **1** Yes **Guideline:** Do not change the default the first time you run IKYSETUP, but change it before going into a production environment.	No	`0` (no)
`unix_sec`	Specifies whether to set up z/OS UNIX level security. (See "Deciding the value of unix_sec" on page 47 for a definition of z/OS UNIX level security.) The value can be: **0** Do not set up - default **1** Already set up **2** Add this level of security **Rule:** If you are changing `unix_sec` to 1 or 2, you must update the `bpx_userid.` and `pgmcntl_dsn.` rows. **Guideline:** Do not set `unix_sec=2` the first time you are running IKYSETUP.	For `unix_sec=2`, the names of the load libraries need to change.	`0` (no)

Variables you can optionally change

To help in filling out the next table of variables (Table 19 on page 55), see "Specifying when the CA certificate and web server certificates expire" on page 54.

Specifying when the CA certificate and web server certificates expire

By default, IKYSETUP creates a CA certificate that expires in 20 years and a web server SSL certificate that expires in five years from the date when the IKYSETUP script is run. If these expiration times are compliant with your security guidelines, no changes are needed to the `ca_exyears`, `ca_expires`, `web_exyears`, and `web_expires` variables in Table 19 on page 55.

You can shorten or extend the lifetime of the CA certificate by altering the value of the `ca_exyears` variable. This variable specifies the lifetime of the CA certificate in years. The default value is 20. You can shorten or extend the lifetime of the web server certificate by altering the value of the `web_exyears` variable. The default value is 5. These variables are listed in IKYSETUP in "Part 3", in the subsection titled "Method 1". Ensure that the value for `web_exyears` is less than the value of `ca_exyears`. If it is not, the web server certificate might be added to the RACF database with the NOTRUST option.

If your security guidelines require that the CA certificate and web server certificate expire at specific dates, you can set these expiration dates in IKYSETUP. Ensure that the web server certificate expires before the date that the CA certificate expires. If you do not, the web server certificate might be added to the RACF database with the NOTRUST option.

Steps for setting the expiration dates for the CA certificate and web server certificate
Perform the following steps to set the expiration dates for the CA certificate and web server certificate in IKYSETUP.

Procedure

1. Comment out all instructions in "Part 3, Method 1" of IKYSETUP. These are the instructions that calculate expiration dates by adding the life spans specified in the `ca_exyears` and `web_exyears` variables to the date that IKYSETUP runs. You must disable these instructions to set the expiration dates to specific dates. ____
2. Remove the comment delimiters from the sample instructions that are listed in "Part 3, Method 2" of IKYSETUP. These are the instructions that set the expiration dates to specific date values. You must enable these instructions.
3. Change the value of the `ca_expires` variable to the date when the CA certificate should expire, in the format *yyyy/mm/dd*. ____
4. Change the value of the `web_expires` variable to the date when the web server certificate should expire, in the format *yyyy/mm/dd*. Specify a date that is before the date specified by the `ca_expires` variable. ____
5. Save your changes. ____

Results

When you are done, you have set the CA certificate expiration date and the web server certificate expiration date that takes effect the next time that you run IKYSETUP.

Table of IKYSETUP variables you can optionally change

Review the values of the variables in Table 19 on page 55 to determine if you want to change any of the defaults in the rightmost column. If you decide to change any value, cross out the default in the rightmost column and record your company's information.

Table 19. IKYSETUP variables you can optionally change

Variable name	Description	Referenced elsewhere	Default value or your company's information
`backup_dsn`	The data set that contains a backup copy of the PKI Services certificate and private key.	No	When you also set *ca_domain*: `'`***`daemon.ca_domain`***`.KEY.BACKUP.P12BIN'` When you do not set *ca_domain*: `'`***`daemon`***`.KEY.BACKUP.P12BIN'` **Note:** The *daemon* refers to the `daemon` variable in this table.

Table 19. IKYSETUP variables you can optionally change (continued)

Variable name	**Description**	**Referenced elsewhere**	**Default value or your company's information**
`ca_domain`	The unique name for the CA when you establish multiple PKI Services CAs. If specified, the first eight characters must uniquely identify the CA. The characters of the `CA_domain` value are limited to the following character set: alphanumeric characters (`a - z`, `A - Z`, `0 - 9`) and the hyphen (`-`). In addition, the first character must not be a number or hyphen.	No	"" **Guideline:** Do not change the default (null) value until you perform advanced customization. (See "Adding a new CA domain" on page 302.)
`ca_expires`	The date that the PKI Services CA certificate expires. By default, IKYSETUP calculates the CA certificate expiration date based on the value of `ca_exyears`. For information about setting this variable, see "Specifying when the CA certificate and web server certificates expire" on page 54.	No	`2030/01/01` The date format is *yyyy*/*mm*/*dd*.
`ca_exyears`	The life span of the PKI Services CA certificate, expressed in years. By default, IKYSETUP calculates the expiration date for the CA certificate by adding the number of years specified in `ca_exyears` to the date that IKYSETUP is run. For information about setting this variable, see "Specifying when the CA certificate and web server certificates expire" on page 54.	No	20

Table 19. IKYSETUP variables you can optionally change (continued)

Variable name	**Description**	**Referenced elsewhere**	**Default value or your company's information**
`ca_ring`	The name of the PKI Services SAF key ring.	`pkiserv.conf` **SAF** `KeyRing` value	When you also set *ca_domain*: `CAring`***.ca_domain*** When you do not set *ca_domain*: `CAring`

`cacert_dsn`	The data set that contains the PKI Services certificate to assist the backup process.	No	When you also set *ca_domain*:
			 `'`***`daemon.ca_domain`***`.CACERT.DERBIN'`
			 When you do not set *ca_domain*:
			 `'`***`daemon`***`.CACERT.DERBIN'`
			 Note: *daemon* refers to the daemon variable in this table.

`caStore`	The name of the PKI Services PKCS #11 token	No	When you also set `ca_comain` or daemon: *daemon*.CATOKEN.*ca_domain* When you do not set `ca_comain` or daemon: CATOKEN
`daemon`	The PKI Services daemon user ID. If you also set *ca_domain*, you can choose to assign a unique user ID to the daemon for each CA domain. **Example:** For a *ca_domain* called BankA, you might choose user ID PKISRVDA.	`pkiserv.conf` **SAF** `KeyRing` value	`PKISRVD`

`export_dsn`	The data set that contains the web server's root CA certificate for copying to file system.	No	When you also set *ca_domain*:
			 `'`***`daemon.ca_domain`***`.WEBROOT.DERBIN'`
			 When you do not set *ca_domain*:
			 `'`***`daemon`***`.WEBROOT.DERBIN'`
			 Note: *daemon* refers to the daemon variable in this table.

Table 19. IKYSETUP variables you can optionally change (continued)

Variable name	**Description**	**Referenced elsewhere**	**Default value or your company's information**
`log_dsn`	The log data set name.	No	When you also set *ca_domain*: `'`***`your-id.ca_domain`***`.IKYSETUP.LOG'` When you do not set *ca_domain*: `'`***`your-id`***`.IKYSETUP.LOG'` **Notes:** 1. The *your-id* refers to the RACF ID of the person running IKYSETUP. (You do not need to add this; MVS adds this for you.) 2. Changing the default is not suggested.
`pkigroup`	The PKI Services administration group. This is a RACF group containing the list of user IDs that are authorized to use PKI Services administration functions. If you also set *ca_domain*, you can choose to assign a unique group name to the administration group for each CA domain. **Example:** For a *ca_domain* called BankA, you might choose group name PKIGRPA.	No	`PKIGRP`
`pkigroup1`, `pkigroup2`	PKI Services administrative groups for granular control of administrative functions.	No	`PKIGRP1`, `PKIGRP2`
`ra_backup_dsn`	The data set that contains a backup copy of the PKI Services RA certificate and private key. This name should be similar but not identical to the `backup_dsn` value.	No	When you also set *ca_domain*: `'`***`daemon.ca_domain`***`.RAKEY.BACKUP.P1 2BIN'` When you do not set *ca_domain*: `'`***`daemon`***`.RAKEY.BACKUP.P12BIN'` **Note:** The *daemon* refers to the `daemon` variable in this table.

Table 19. IKYSETUP variables you can optionally change (continued)

Variable name	Description	Referenced elsewhere	Default value or your company's information
`signing_ca_label`	The label of the CA certificate that is the superior (signer) of the PKI Services CA. If specified, the value must match the label of an existing CERTAUTH certificate in RACF that has a private key. Use this value to create a CA hierarchy when you establish multiple PKI Services CAs.	No	""
`surrog`	The surrogate user ID for PKI Services. If you also set *ca_domain*, you can choose to assign a unique user ID as the surrogate user ID for each CA domain. **Example:** For a *ca_domain* called BankA, you might choose user ID PKISERVA. **Note:** This cannot be an existing user ID (because IKYSETUP creates the user ID with the NOPASSWORD attribute).	Surrogate user ID in `httpd*.conf`	`PKISERV`
`vsamhlq`	The high-level qualifier of the VSAM data sets for PKI Services. **Note:** The RACF administrator gets this information from the MVS programmer.	• ObjectStore `*DSN` values in `pkiserv.conf` • Data sets names in IKYCVSV1	Same as the `daemon` variable earlier in this table.
`web_expires`	The date that the web server certificate expires. By default, IKYSETUP calculates the web server certificate expiration date based on the value of `web_exyears`. For information about setting this variable, see "Specifying when the CA certificate and web server certificates expire" on page 54.	No	`2015/01/01` The date format is *yyyy*/*mm*/*dd*.

Table 19. IKYSETUP variables you can optionally change (continued)

Variable name	Description	Referenced elsewhere	Default value or your company's information
`web_exyears`	The life span of the web server certificate, expressed in years. By default, IKYSETUP calculates the expiration date for the web server certificate by adding the number of years specified in `web_exyears` to the date when IKYSETUP is run. For information about setting this variable, see "Specifying when the CA certificate and web server certificates expire" on page 54.	No	5
`web_label`	The label for the web server's certificate.	No	`SSL Cert`
`webserver`	The web server's daemon user ID.	See web server documentation.	`WEBSRV`

Steps for performing RACF tasks using IKYSETUP

Use the following directions to run IKYSETUP only if you have not done so for a previous release (or if you are changing values).

You can use the following directions to run IKYSETUP with minimal changes or to extensively customize it.

Guideline: If this is your first attempt to use IKYSETUP, change only the IKYSETUP variables in the section `Things you must change`. You can refine IKYSETUP later, after you are familiar with the process of updating and running it.

The following flowchart illustrates the iterative nature of the process of updating IKYSETUP:

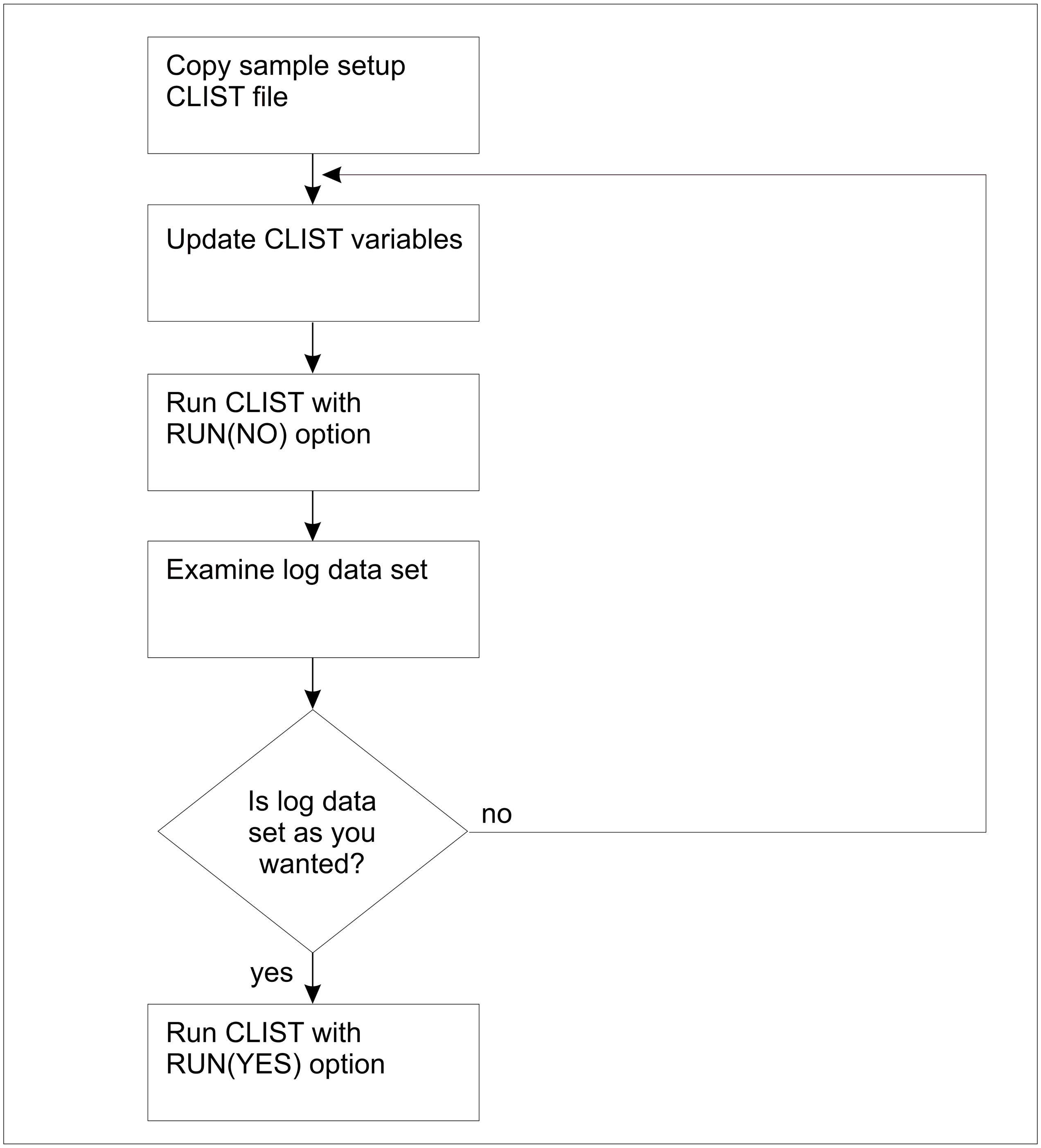

Copy sample setup CLIST file
Update CLIST variables
Run CLIST with RUN(NO) option
Examine log data set
Is log data set as you wanted?
no
yes
Run CLIST with RUN(YES) option

Figure 2. Flowchart of the process of updating IKYSETUP

Perform the following steps to use IKYSETUP to perform RACF administration tasks:

1. Copy `SYS1.SAMPLIB(IKYSETUP)` to a data set you are permitted to edit.

   ____

2. Edit the IKYSETUP code to update the values of variables you changed in Table 11 on page 41.

   The following example shows how to change the `pkigroup_mem.` variables. (Remember that for `pkigroup_mem.`, you set `pkigroup_mem.0` to the number of items in the list and `pkigroup_mem.1` through `pkigroup_mem.n` to the PKI Services administration group member IDs.)

Example:

```
pkigroup_mem.0=3       /* Number of pkigroup members to connect */
pkigroup_mem.1="TOM"
pkigroup_mem.2="DICK"
pkigroup_mem.3="HARRY"
```

____

3. If necessary, update the values of variables you changed in Table 18 on page 50.

   The following example shows how to change the `key_type` variable.

   Example:

   ```
   key_type=1
   ```

____

4. Optionally update any variables you changed in Table 19 on page 55.

   The following example shows how to change the `log_dsn` variable.

   Example:

   ```
   log_dsn="PRIVATE.IKYSETUP.LOG"
   ```

   If you are changing the values of `ca_exyears`, `ca_expires`, `web_exyears`, or `web_expires`, see "Specifying when the CA certificate and web server certificates expire" on page 54 for instructions.

____

5. Run IKYSETUP by entering the following command:

   ```
   EX 'data-set-name(IKYSETUP)' 'RUN(NO)'
   ```

   Notes:

   a. The user ID that runs IKYSETUP must be a RACF SPECIAL user ID.
   b. When IKYSETUP runs, it prompts you to enter your secret passphrase. (This is for encrypting the backup copy of your CA certificate and private key.) Be aware that asterisks do not replace the secret passphrase; it appears on the screen in the clear.

      Important: Make a note of this passphrase. If you forget it, your backup is useless.
   c. The NO option in the command specifies displaying the commands only. (This creates a log data set listing the commands and other information. Alternative parameters are: YES, which indicates running IKYSETUP as is, and PROMPT, indicates prompting the user before running each command.)

____

6. Review the log data set. (See "Sample IKYSETUP log data set" on page 63 for an example of the data that appears on your display when you are running IKYSETUP; this is similar to the contents of the log data set.) The first part identifies the tasks and shows the commands that run to perform those tasks. Review this to ensure that the issued commands match your expectations. (For more information about these commands, see "Actions IKYSETUP performs by issuing RACF commands" on page 619.) The bottom part provides a record of important information that you need for later steps, such the name of your daemon user ID. Review this information to ensure that the values are the ones you want.

   If you want to change any of the commands or information in the log data set, you need to change additional values in IKYSETUP. Remember to record any additional changes in Table 11 on page 41, Table 18 on page 50, and Table 19 on page 55. Then go back to Step "3" on page 62.

____

7. If the log data set includes the commands and information you want, rerun the IKYSETUP code by entering the following command:

   ```
   EX 'data-set-name(IKYSETUP)' 'RUN(YES)'
   ```

   ____

8. After running IKYSETUP with `RUN(YES)`, examine the results recorded in the log data set. Investigate and rerun (potentially by hand) any failing commands. Investigate informational messages and make any necessary corrections. (Informational messages usually indicate a setup problem that might affect operations later. For example, any informational message from the RACDCERT commands that indicate that the certificate has been marked `NO` TRUST is an error.)

   ____

9. If you intend to use encrypted LDAP passwords, you need to perform additional RACF administration tasks; see "Using encrypted passwords for LDAP servers" on page 501.

   ____

Sample IKYSETUP log data set

Here is an example of the data that appears when you run IKYSETUP.

```
Creating users and groups ...
ADDUSER  PKISRVD name('PKI Srvs Daemon')  nopassword  omvs(uid(554)  assize(256000000)
threads(512))
ADDUSER  PKISERV nopassword   omvs(uid(555))  name('PKI Srvs Surrogate')
ADDGROUP  PKIGRP OMVS(GID(655))
SETROPTS EGN GENERIC(DATASET)
ADDSD 'PKISRVD.**' UACC(NONE)
PERMIT 'PKISRVD.**' ID(PKISRVD) ACCESS(ALTER)

Allowing administrators to access PKI VSAM databases ...
PERMIT 'PKISRVD.**' ID(PKIGRP) ACCESS(CONTROL)
SETROPTS GENERIC(DATASET) REFRESH

Creating the CA certificate ...
RACDCERT GENCERT CERTAUTH SUBJECTSDN(OU('Human Resources Certificate Authority')
O('Your Company') C('Your Country 2 Letter Abbreviation'))  WITHLABEL('Local PKI CA')
NOTAFTER(DATE(2033/06/14))  SIZE(2048)

Backing up the CA certificate ...
RACDCERT CERTAUTH EXPORT(LABEL('Local PKI CA'))  DSN('PKISRVD.KEY.BACKUP.P12BIN')
FORMAT(PKCS12DER)
PASSWORD('******')

Marking CA certificate as HIGHTRUST ...
RACDCERT CERTAUTH ALTER(LABEL('Local PKI CA')) HIGHTRUST

Saving the CA certificate to a data set ...
RACDCERT CERTAUTH EXPORT(LABEL('Local PKI CA'))  DSN('PKISRVD.CACERT.DERBIN') FORMAT(CERTDER)

Creating the RA certificate ...
RACDCERT ID(PKISRVD) GENCERT SUBJECTSDN(CN('Registration Authority')
OU('Human Resources Certificate Authority')
O('Your Company') C('Your Country 2 Letter Abbreviation'))
KEYUSAGE(HANDSHAKE) SIGNWITH(CERTAUTH LABEL('Local PKI CA'))
NOTAFTER(DATE(2033/06/14)) WITHLABEL('Local PKI RA')

Backing up RA certificate ...
RACDCERT ID(PKISRVD) EXPORT(LABEL('Local PKI RA'))  DSN('PKISRVD.RAKEY.BACKUP.P12BIN')
FORMAT(PKCS12DER)  PASSWORD('******')

Creating the PKI Services keyring ...
RACDCERT ADDRING(CAring) ID(PKISRVD)
RACDCERT ID(PKISRVD) CONNECT(CERTAUTH  LABEL('Local PKI CA')  RING(CAring) USAGE(PERSONAL)
DEFAULT)
RACDCERT ID(PKISRVD) CONNECT(LABEL('Local PKI RA')  RING(CAring) USAGE(PERSONAL))

Creating the Webserver SSL certificate and keyring ...
RACDCERT GENCERT ID(WEBSRV) SIGNWITH(CERTAUTH  LABEL('Local PKI CA'))  WITHLABEL('SSL Cert')
SUBJECTSDN(CN('www.YourCompany.com') O('Your Company') L('Your City')
SP('Your Full State or Province Name') C('Your Country 2 Letter Abbreviation'))
NOTAFTER(DATE(2018/06/14))
RACDCERT ADDRING(SSLring) ID(WEBSRV)
```

```
RACDCERT  ID(WEBSRV) CONNECT(ID(WEBSRV)  LABEL('SSL Cert') RING(SSLring) USAGE(PERSONAL)
DEFAULT)
RACDCERT  ID(WEBSRV) CONNECT(CERTAUTH  LABEL('Local PKI CA') RING(SSLring))

Saving the webserver's root CA certificate to a  data set for OPUT ...
RACDCERT CERTAUTH EXPORT(LABEL('Local PKI CA'))  DSN('PKISRVD.WEBROOT.DERBIN') FORMAT(CERTDER)

Giving PKISRVD access to BPX.SERVER ...
RDEFINE FACILITY BPX.SERVER
PERMIT BPX.SERVER CLASS(FACILITY)  ID(PKISRVD) ACCESS(READ)

Allowing the PKI Services daemon to act as a CA ...
RDEFINE FACILITY IRR.DIGTCERT.GENCERT
RDEFINE FACILITY IRR.DIGTCERT.LISTRING
PERMIT IRR.DIGTCERT.GENCERT CLASS(FACILITY)  ID(PKISRVD) ACCESS(CONTROL)
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY)  ID(PKISRVD) ACCESS(READ)

Allowing the Webserver to access its keyring ...
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY)  ID(WEBSRV) ACCESS(READ)

Allowing the Webserver to switch identity to PKISERV ...
SETROPTS CLASSACT(SURROGAT)
RDEFINE SURROGAT BPX.SRV.PKISERV
PERMIT BPX.SRV.PKISERV CLASS(SURROGAT)  ID(WEBSRV) ACCESS(READ)
SETROPTS RACLIST(SURROGAT) REFRESH

Allowing the PKI Services daemon to use ICSF ...
SETROPTS  GENERIC(CSFKEYS CSFSERV)
SETROPTS  GENERIC(CSFKEYS CSFSERV) REFRESH
RDEFINE CSFKEYS IRR.DIGTCERT.CERTIFAUTH.* UACC(NONE)
PERMIT IRR.DIGTCERT.CERTIFAUTH.* CLASS(CSFKEYS)  ID(PKISRVD) ACCESS(READ)
SETROPTS CLASSACT(CSFKEYS) RACLIST(CSFKEYS)
SETROPTS RACLIST(CSFKEYS) REFRESH

Creating the STARTED class profile for the daemon ...
RDEFINE STARTED PKISERVD.* STDATA(USER(PKISRVD))
SETROPTS CLASSACT(STARTED) RACLIST(STARTED)
SETROPTS RACLIST(STARTED) REFRESH

Allowing PKISERV to request certificate functions ...
SETR GENERIC(FACILITY)
RDEFINE FACILITY IRR.RPKISERV.**
PERMIT IRR.RPKISERV.** CLASS(FACILITY)  ID(PKISERV) ACCESS(CONTROL)

Creating the profile to protect PKI Admin functions ...
RDEFINE FACILITY IRR.RPKISERV.PKIADMIN
PERMIT IRR.RPKISERV.PKIADMIN CLASS(FACILITY)  ID(PKIGRP) ACCESS(UPDATE)
PERMIT IRR.RPKISERV.PKIADMIN CLASS(FACILITY)  ID(PKISERV) ACCESS(NONE)
SETROPTS RACLIST(FACILITY) REFRESH
-------------------------------------------------
Information needed for PKI Services UNIX set up:
-------------------------------------------------

The daemon user ID is:
  PKISRVD

The VSAM high level qualifier is:
  PKISRVD
This is needed for the [ObjectStore] section in pkiserv.conf

The PKI Services' DER encoded certificate is in data set:
  'PKISRVD.CACERT.DERBIN'

The webserver's DER encoded root
CA certificate is in data set:
  'PKISRVD.WEBROOT.DERBIN'
This must be OPUT to /var/pkiserv/cacert.der with  the BINARY option

The fully qualified PKI Services' SAF keyring is:
  PKISRVD/CAring
This is needed for the [SAF] section in pkiserv.conf

The label of the PKI Services' RA certificate is:
  Local PKI RA
This is needed for the [SAF] section in pkiserv.conf

The PKI Services CA DN is:
  OU=Human Resources Certificate Authority,O=Your Company,C=Your Country 2 Letter Abbreviation
The suffix must match the LDAP suffix in slapd.conf

The PKI Services RA DN is:
  CN=Registration Authority,OU=Human Resources Certificate Authority,O=Your Company,
C=Your Country 2 Letter Abbreviation
The suffix must match the LDAP suffix in slapd.conf

The recommended location for the pkiserv.conf and pkiserv.tmpl is:
  /etc/pkiserv

Set the following environment variables in pkiserv.envars:
  _PKISERV_CONFIG_PATH=/etc/pkiserv

Set the following environment variable in your httpd envvars files:
  _PKISERV_CONFIG_PATH=/etc/pkiserv

The webserver's SAF keyring is:
  SSLring
This is needed for the KeyFile directive in virtual host files

The Webserver's DN is:
  CN=www.YourCompany.com,O=Your Company,L=Your City,ST=Your Full State or Province Name,
C=Your Country 2 Letter Abbreviation
The left most RDN must be the webserver's fully qualified domain name
```

Chapter 5. Configuring the UNIX runtime environment

You need to perform all of the tasks in this topic if you are configuring PKI Services for the first time. If you have already configured PKI Services for an earlier release, you might need to perform some of the tasks in this topic if you are adding support for a function.

After the RACF administrator performs the tasks necessary to set up PKI Services, the UNIX programmer needs to perform the following tasks:

- If necessary, copy files.
- If necessary, update the environment variables file.
- If necessary, update the configuration file.
- If configuring PKI Services for the first time or adding a new CA domain, set up the `/var/pkiserv` directory.

The following table summarizes information about copying and updating files. To view the contents of any of these files, see Chapter 31, "Other code samples," on page 649.

Table 20. Deciding which files to copy and change

File	Purpose	Need to copy?	Need to change?
`expiringmsg.form`	The form for an email sent to a user when a certificate is going to expire.	Only if your company sends an email notification to a user about a certificate that is going to expire	**Guideline:** Make no changes to this file until later. See "Customizing email notifications sent to users" on page 317 for details about making changes.
`pendingmsg.form`	The form for an email sent to an administrator when requests are pending approval.	Only if your company sends an email notification to an administrator about requests that are pending approval.	**Guideline:** Make no changes to this file until later. See "Customizing email notifications sent to users" on page 317 for details about making changes.
`pendingmsg2.form`	The form for an email sent to an administrator when requests are approved with modifications.	Only if your company sends an email notification to an administrator about requests approved with modifications.	**Guideline:** Make no changes to this file until later. See "Customizing email notifications sent to users" on page 317 for details about making changes.

Table 20. Deciding which files to copy and change (continued)

File	Purpose	Need to copy?	Need to change?
`pkiserv.conf`	Configuration file. Contains various settings and values PKI Services needs.	Only if you are configuring PKI Services for the first time.	The UNIX programmer might need to change the **LDAP** section of this file. **Guideline:** Do not change it now but change it later when you perform "Steps for tailoring the LDAP section of the configuration file" on page 108. The UNIX programmer might need to update the non-LDAP section of the `pkiserv.conf` configuration file when you add support for a function. For more information, see "Optionally updating the pkiserv.conf configuration file" on page 72.
`pkiserv.envars`	The environment variables file.	Only if you are configuring PKI Services for the first time and the file needs changes.	UNIX programmer might have to update this file. See "Optionally updating PKI Services environment variables" on page 70.
`pkiserv.tmpl`	The certificate templates file used with REXX CGI execs. It contains HTML-style code that builds the web pages underlying certificate requests.	Only if you are configuring PKI Services for the first time and using the REXX CGI execs to implement the PKI Services web application.	**Guideline:** Make no changes to this file until later. See Chapter 11, "Customizing the end-user web application if you use REXX CGI execs," on page 135 for details about making changes.
`PKIServ.xsd`	The XML schema that defines the syntax of the XML certificate templates file `pkitmpl.xml`.	Only if you are configuring PKI Services for the first time and using the JavaServer pages (JSPs) to implement the PKI Services web application.	**Rule:** Do not make changes to this file.
`pkitmpl.xml`	The certificate templates file used with JavaServer pages (JSPs). It defines applications and certificates in XML.	Only if you are configuring PKI Services for the first time and using the JSPs to implement the PKI Services web application.	**Guideline:** Make no changes to this file until later. See Chapter 13, "Implementing the web application using JavaServer pages," on page 237 for details about making changes.

Table 20. Deciding which files to copy and change (continued)

File	Purpose	Need to copy?	Need to change?
`readymsg.form`	The form for an email sent to a user when the PKI Services administrator has approved a certificate request and the certificate is ready for retrieval.	Only if your company sends an email notification to a user after the PKI Services administrator has approved a certificate request and the certificate is ready for retrieval.	**Guideline:** Make no changes to this file until later. See "Customizing email notifications sent to users" on page 317 for details about making changes.
`rejectmsg.form`	The form for an email sent to a user when the PKI Services administrator has rejected a certificate request.	Only if your company sends an email notification to a user after the PKI Services administrator has rejected a certificate request.	**Guideline:** Make no changes to this file until later. See "Customizing email notifications sent to users" on page 317 for details about making changes.
`renewcertmsg.form`	The form for an email sent to a user when PKI Services has automatically renewed an expiring certificate.	Only if your company enables automatic renewal of certificates.	**Guideline:** Make no changes to this file until later. See "Customizing email notifications sent to users" on page 317 for details about making changes.
`recoverymsg.form`	The form for an email sent to a user who has requested that PKI Services recover a certificate for which PKI Services generated the key pair.	Only if your company allows users to request that PKI Services generate key pairs for certificate requests.	**Guideline:** Make no changes to this file until later. See "Customizing email notifications sent to users" on page 317 for details about making changes.

Steps for copying files

Before you begin

- You need to obtain the following document:
  - *z/OS UNIX System Services Planning*
- You need to know the file system directory where the MVS programmer installed PKI Services and the runtime directory, `install-dir` and `runtime-dir` in the commands that follow. The defaults are `/usr/lpp/pkiserv/` and `/etc/pkiserv`, in that order. The MVS programmer was asked to record any changes to these defaults; see Table 3 on page 11.
- The user ID you use for copying files must have superuser authority.

Perform the following steps to copy the files:

1. If you are configuring PKI Services for the first time, copy the configuration file by entering the following command from the UNIX command line:

   ```
   cp -p /install-dir/samples/pkiserv.conf runtime-dir
   ```

   ____

2. If you are configuring PKI Services for the first time, copy the templates files. Do either Step "2.a" on page 70 or Step "2.b" on page 70.

a. If you are using REXX CGIs to implement the PKI Services web application, copy the text template file by entering the following command from the UNIX command line:

```
cp -p /install-dir/samples/pkiserv.tmpl runtime-dir
```

b. If you are using JavaServer pages (JSPs) to implement the PKI Services web application, copy the XML template file and the XML schema file by entering the following commands from the UNIX command line:

```
cp -p /install-dir/samples/pkitmpl.xml runtime-dir
cp -p /install-dir/samples/PKIServ.xsd runtime-dir
```

___

3. If your company is sending email notifications to users (when certificate requests are rejected or when certificates are ready for retrieval or expiring), copy the appropriate notification files from the samples directory to the runtime directory. For example:

```
cp -p /install-dir/samples/rejectmsg.form runtime-dir
cp -p /install-dir/samples/readymsg.form runtime-dir
cp -p /install-dir/samples/expiringmsg.form runtime-dir
```

___

4. If your company is sending email notifications to administrators when certificate requests are pending approval, copy the `pendingmsg.form` notification file from the samples directory to the runtime directory. For example:

```
cp -p /install-dir/samples/pendingmsg.form runtime-dir
```

___

5. If your company is sending email notifications to administrators when certificate requests are approved with modifications, copy the `pendingmsg2.form` notification file from the samples directory to the runtime directory. For example:

```
cp -p /install-dir/samples/pendingmsg2.form runtime-dir
```

___

6. If your company allows users to request that PKI Services create key pairs (private key and public key) for certificate requests, copy the notification file that is used when a user requests that PKI Services recover a certificate for which it created the keys. For example:

```
cp -p /install-dir/samples/recoverymsg.form runtime-dir
```

___

7. If you are configuring PKI Services for the first time, examine the values in the environment variables file (by default, `pkiserv.envars`). If any values need to change, copy this file by entering the following command:

```
cp -p /install-dir/samples/pkiserv.envars runtime-dir
```

___

Optionally updating PKI Services environment variables

You need to perform this task only if any one of the following conditions is true:

- You are configuring PKI Services for the first time.

- You are adding an additional CA domain.
- You want to send email notifications (for rejected certificate requests or certificates that are ready for retrieval or expiring) and you did not use the default location for sendmail (`/bin/sendmail`).
- You intend to use automatic certificate renewal.
- You are implementing an **autorenew** exit.
- You intend to use JavaServer pages (JSPs) instead of REXX CGIs for the PKI Services web pages.
- You intend to run PKI Services in FIPS mode.

You need to define certain environment variables (such as LIBPATH) for the PKI Services daemon to run. There are two files related to environment variables:

- A sample environment variables file, `pkiserv.envars` (by default in `/usr/lpp/pkiserv/samples/`)
- `SYS1.PROCLIB` member PKISERVD (You can use the ENVAR parameter to point to the environment variables file.)

You can use `pkiserv.envars` to set environment variables for the PKI Services daemon. This file contains most of the environment variables needed to run the daemon.

You need to change the file if you did not use the default for any of these things:

- The install directory for PKI Service (`/usr/lpp/pkiserv`)
- The message level
- The location for sendmail (`/bin/sendmail`)

Guideline: If you need to make changes to the `pkiserv.envars` file, copy the file to another directory (such as `/etc/pkiserv`) and make changes only to the copy.

PKISERVD is the sample procedure to start PKI Services. (For sample code, see "PKISERVD sample procedure to start PKI Services daemon" on page 682.) PKISERVD sets the **TZ** (time zone) environment variable because it is very likely that the value of this variable needs to change. PKISERVD also includes parameters specifying the directory containing the environment variables file (DIR) and the file name of the environment variables file (FN). If you make a copy of `pkiserv.envars` as suggested, you also need to change the name of the directory in PKISERVD (for example, `DIR="/etc/pkiserv"`) and possibly the file name (for example, `FN="pki.env"`).

Note: You can change all of the following on the START command:

- Environment variables directory
- File name
- Job output class
- Region size
- Standard output
- Standard error
- Time zone

See "Steps for starting the PKI Services daemon" on page 129.

Because of the limitation of the number of characters allowed in the PARM=*operand* on the JCL EXEC card, take care to ensure that the total length of the environment variables directory and file name, **TZ** value, and `stdout` and `stderr` redirection values do not exceed the 100 character maximum.

You must specify any environment variables that PKI Services requires either in the PKISERVD procedure or in the environment variables file (`pkiserv.envars`). **Guideline:** Make your additions and changes to the environment variables file, rather than to the PKISERVD procedure.

(Optional) Steps for updating PKI Services environment variables

Before you begin

Examine the values in the environment variables file (by default, `pkiserv.envars`) and determine whether you need to update any values. (See "Environment variables in the environment variables file" on page 615 for a description of the environment variables and "The pkiserv.envars environment variables file" on page 617 for a code sample of the environment variables file.)

Procedure

Perform the following steps to update PKI Services environment variables:

1. If you did not install sendmail in its default location (`/bin`), update the PATH environment variable to the location of sendmail.
2. Make any needed changes to PKISERVD, such as updating the path name of the environment variables file (FN and DIR parameters). (See "PKISERVD sample procedure to start PKI Services daemon" on page 682 for a code sample of the PKISERVD procedure.)

   ____

3. _PKISERV_VARDIR specifies the path name for a directory in which PKI Services writes persistent data. The maximum length of the path name is 256 characters, including the trailing /. The default value (if you do not set the environment variable) is `/var/pkiserv`. For example:

   ```
   _PKISERV_VARDIR=/var/mypkiserv/
   ```

   ____

4. If you do not define _PKISERV_EXIT or if it contains a null value, the PKI exit processing is disabled. If you have implemented an AUTORENEW exit, set _PKISERV_EXIT to the absolute path name of the exit program name. The maximum length is 256 characters including the program name. For example:

   ```
   _PKISERV_EXIT=/mydir/renewexit
   ```

   ____

5. If you want to run PKI Services in FIPS mode, update _PKISERV_FIPS_LEVEL to the desired FIPS level. For example:

   ```
   _PKISERV_FIPS_LEVEL=1
   ```

   Prior to changing this environment variable, make sure all of the requests have a status of `Completed`. Otherwise, certificates may not be issued for pending or approved requests that are not compliant with the new FIPS level.

   For a description of the _PKISERV_FIPS_LEVEL environment variable and the appropriate values, see "Environment variables in the environment variables file" on page 615.

   ____

6. Update any other values as necessary.

   ____

Optionally updating the pkiserv.conf configuration file

You need to update the `pkiserv.conf` configuration file if you meet any of the following conditions:

- You are configuring PKI Services for the first time
- You are adding support for:
  - Running multiple instances of PKI Services in a sysplex.
  - Running multiple CA domains on a single z/OS image. (See "Adding a new CA domain" on page 302.)

  - Sending email notifications to users if the PKI Services administrator rejects certificate requests or certificates are ready for retrieval or expiring
  - Customizing certificate revocation list (CRL) distribution point processing. (See "Customizing distribution point CRLs" on page 290 for details.)
  - Automatic renewal of expiring certificates
  - Sending email notifications to administrators if any requests are pending approval
  - A timeout value for the PKI Services exit.
  - Generation of key pairs (public and private key) for certificates
  - Setting the time that the daily maintenance task runs, or the days that it runs, or specifying that it is not to run when the PKI Services daemon starts
  - The certificate management protocol (CMP)
  - Using Db2 tables instead of VSAM files for the object store and ICL.
  - Creating CRLs without the Issuing Distribution Point extension.
  - Constraining the CA path length.
  - Granular control of administrative functions.
  - WTO notification.
  - SCEP request enhancement.
  - Enrollment over Secure Tunneling protocol (EST)
  - RSASSA-PSS signature algorithm
- You installed a new release of z/OS and had configured PKI Services on the earlier release. (For more information see "Updating pkiserv.conf after installing a new release of z/OS" on page 94.

You can also optionally update the file if you want to change certain default values.

The `pkiserv.conf` configuration file for the PKI Services daemon consists of sections of name-value pairs. **Important:** Everything in the `pkiserv.conf` file, including section names, keys, and values, is case-sensitive.

Each section of the `pkiserv.conf` configuration file has a title enclosed in square brackets. The configuration file includes the following sections:

[OIDs]
The OIDs section specifies the object identifiers for various nicknames PKI Services uses internally. The OIDs are specified in the following form:

```
name=dotted-decimal
```

The following excerpt is from the OIDs section:

```
[OIDs]
⋮
MyPolicy=1.2.3.4
```

[ObjectStore]
The ObjectStore section specifies operational information for the object store and issued certificate list (ICL).

The following excerpt is from the ObjectStore section:

```
[ObjectStore]
ObjectDSN='pkisrvd.vsam.ost'
⋮
```

[CertPolicy]
The CertPolicy section is for CA policy information.

The following excerpt is from the CertPolicy section:

```
[CertPolicy]
SigAlg1=sha-256WithRSAEncryption
⋮
```

[General]

The General section is for general information.

The following excerpt is from the General section:

```
[General]
InitialThreadCount=10
⋮
```

[SAF]

The SAF section is for information about the SAF (RACF) key ring that is used for CA certificate and private key storage.

The following excerpt is from the SAF section:

```
[SAF]
KeyRing=PKISRVD/CAring
```

[LDAP]

The LDAP section contains information about the LDAP server for posting certificates and CRLs.

The following excerpt is from the LDAP section:

```
[LDAP]
NumServers=1
⋮
```

The UNIX programmer needs to update the **LDAP** section of this file. **Guideline:** Do not change it now but change it later when you perform "Steps for tailoring the LDAP section of the configuration file" on page 108.

(Optional) Steps for updating the configuration file

Before you begin

The following table provides information about parameters in the `pkiserv.conf` configuration file. (It omits parameters for the LDAP section. For information about these parameters, see Table 25 on page 108.) Read the parameter descriptions, and examine the values that are provided in the sample configuration file, which is shown in the rightmost column to ensure that the values meet your company's requirements. As necessary, cross out the sample values and enter the information appropriate to your own organization's needs and policies.

Table 21. Information needed for updating the configuration file

Parameter	Information needed	Where to get this information	Sample value or your customized value
OIDs section			

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`MyPolicy`	A registered Object ID identifying your organization's usage policy, for example: `1.2.3.4`	If you are creating your own certificate policy, see "Using certificate policies" on page 284 for information about creating certificate policies. Otherwise, do not change this information.	`1.2.3.4` If you need to use the CertificatePolicies extension, replace `1.2.3.4` with the value of your Object ID:
ObjectStore section			
`DBType`	Repository for the object store and issued certificate list (ICL). Valid values are: • `VSAM` • `DB2` The default value is `VSAM`. If `DBType` is `VSAM`, specify values for the parameters `ObjectDSN`, `ObjectTidDSN`, `ObjectStatusDSN`, `ObjectRequestorDSN`, `ICLDSN`, `ICLStatusDSN`, and `ICLRequestorDSN`. If `DBType` is `DB2`, these parameters are ignored. If `DBType` is `DB2`, specify values for the parameters `DBPackage`, `DBSubsystem`, and `DBWaitTime`. If `DBType` is `VSAM`, these parameters are ignored.	UNIX programmer decides this value.	`VSAM`
`DBVersion`	The format version in use by the PKI Services backing storage. The following two format versions are supported: **0** Original PKI Services back storage data format, supported by all PKI Services releases. **1** Extended PKI Services backing storage data format with improved requester and SCEP transaction ID tracking, supported by PKI Services Version 2 Release 3 and later. If the parameter is not specified, a value of `0` is assumed.	Unix programmer decides this value. If you are upgrading an existing PKI Services installation, do not change this value until the software upgrade is complete on this installation and all other installations within the sysplex. For more information, see "Creating the object store and ICL using VSAM data sets" on page 116 and "Creating the object store and ICL using Db2 tables" on page 122.	`0`
`DBPackage`	Name of the Db2 package this instance of PKI Services uses for the object store and ICL in the Db2 subsystem specified by the `DBSubsystem` parameter. If `DBType` is `VSAM`, this parameter is ignored.	Db2 programmer decides this value.	`MasterCA`
`DBSubsystem`	Name of the Db2 subsystem or group attachment that is used by this instance of PKI Services. If `DBType` is `VSAM`, this parameter is ignored.	Db2 programmer decides this value.	`DSN9`

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`DBWaitTime`	Specifies the amount of time that PKI Services will wait for Db2 to restore operations if Db2 should become unavailable while PKI Services is running. If `DBType` is `VSAM`, this parameter is ignored.	Db2 programmer decides this value.	`30m`
`ObjectDSN`	VSAM data set name for the object store base cluster. This is the request database. Each VSAM request record consists of a fixed header followed by a variable-length section. If `DBType` is `DB2`, this parameter is ignored. **Guideline:** If you are adding a new CA domain, insert the *ca_domain* value from Table 19 on page 55 as the second qualifier in the data set name. **Example:** `'pkisrvd.employee.vsam.ost'`	For the high-level qualifier that precedes the period, see the *vsamhlq* variable in Table 19 on page 55. The name of the file, following the period, can change; the MVS programmer who creates the VSAM data sets usually decides these names.	`'pkisrvd.vsam.ost'` Note that this begins with the VSAM high-level qualifier.
`ObjectTidDSN`	VSAM data set name for the object store transaction ID (TID) alternate index. If `DBType` is `DB2`, this parameter is ignored. **Guideline:** If you are adding a new CA domain, insert the *ca_domain* value from Table 19 on page 55 as the second qualifier in the data set name. **Example:** `'pkisrvd.employee.vsam.ost.path'`	For the high-level qualifier that precedes the period, see the *vsamhlq* variable in Table 19 on page 55. The name of the file, following the period, can change; the MVS programmer who creates the VSAM data sets usually decides these names.	`'pkisrvd.vsam.ost.path'` Note that this begins with the VSAM high-level qualifier.
`ObjectStatusDSN`	VSAM data set name for the object store status alternate index. If `DBType` is `DB2`, this parameter is ignored. **Guideline:** If you are adding a new CA domain, insert the *ca_domain* value from Table 19 on page 55 as the second qualifier in the data set name. **Example:** `'pkisrvd.employee.vsam.ost.status'`	For the high-level qualifier that precedes the period, see the *vsamhlq* variable in Table 19 on page 55. The name of the file, following the period, can change; the MVS programmer who creates the VSAM data sets usually decides these names.	`'pkisrvd.vsam.ost.status'` Note that this begins with the VSAM high-level qualifier.
`ObjectRequestorDSN`	VSAM data set name for the object store requestor alternate index. If `DBType` is `DB2`, this parameter is ignored. **Guideline:** If you are adding a new CA domain, insert the *ca_domain* value from Table 19 on page 55 as the second qualifier in the data set name. **Example:** `'pkisrvd.employee.vsam.ost.requestr'`	For the high-level qualifier that precedes the period, see the *vsamhlq* variable in Table 19 on page 55. The name of the file, following the period, can change; the MVS programmer who creates the VSAM data sets usually decides these names.	`'pkisrvd.vsam.ost.requestr'` Note that this begins with the VSAM high-level qualifier.

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`ObjectSCEPTidDSN`	VSAM data set name for the object SCEP transaction ID alternate index. If DBType is DB2, this parameter is ignored. **Guideline:** If you are adding a new CA domain, insert the *ca_domain* value from Table 19 on page 55 as the second qualifier in the data set name. **Example:** `'pkisrvd.employee.vsam.ost.sceptid'`	For the high-level qualifier that precedes the period, see the `vsamhlq` variable in Table 19 on page 55. The name of the file, following the period, can change; the MVS programmer who creates the VSAM data sets usually decides these names.	`'pkisrvd.vsam.ost.sceptid'` Note that this begins with the VSAM high-level qualifier.
`ICLDSN`	VSAM data set name for the ICL base cluster. This data set contains the certificates that have been issued. Each VSAM ICL record consists of a fixed header followed by a variable-length section containing the BER-encoded certificates. If DBType is DB2, this parameter is ignored. **Guideline:** If you are adding a new CA domain, insert the *ca_domain* value from Table 19 on page 55 as the second qualifier in the data set name. **Example:** `'pkisrvd.employee.vsam.icl'`	For the high-level qualifier that precedes the period, see the *vsamhlq* variable in Table 19 on page 55. The name of the file, following the period, can change; the MVS programmer who creates the VSAM data sets usually decides these names.	`'pkisrvd.vsam.icl'` Note that this begins with the VSAM high-level qualifier.
`ICLStatusDSN`	VSAM data set name for ICL status alternate index. If DBType is DB2, this parameter is ignored. **Guideline:** If you are adding a new CA domain, insert the *ca_domain* value from Table 19 on page 55 as the second qualifier in the data set name. **Example:** `'pkisrvd.employee.vsam.icl.status'`	For the high-level qualifier that precedes the period, see the *vsamhlq* variable in Table 19 on page 55. The name of the file, following the period, can change; the MVS programmer who creates the VSAM data sets usually decides these names.	`'pkisrvd.vsam.icl.status'` Note that this begins with the VSAM high-level qualifier.
`ICLRequestorDSN`	VSAM data set name for ICL requestor alternate index. If DBType is DB2, this parameter is ignored. **Guideline:** If you are adding a new CA domain, insert the *ca_domain* value from Table 19 on page 55 as the second qualifier in the data set name. **Example:** `'pkisrvd.employee.vsam.icl.requestr'`	For the high-level qualifier that precedes the period, see the *vsamhlq* variable in Table 19 on page 55. The name of the file, following the period, can change; the MVS programmer who creates the VSAM data sets usually decides these names.	`'pkisrvd.vsam.icl.requestr'` Note that this begins with the VSAM high-level qualifier.

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`ICLSCEPTidDSN`	VSAM data set name for the ICL SCEP transaction ID alternate index. If DBType is DB2, this parameter is ignored. **Guideline:** If you are adding a new CA domain, insert the *ca_domain* value from Table 19 on page 55 as the second qualifier in the data set name. **Example:** `'pkisrvd.employee.vsam.icl.sceptid'`	For the high-level qualifier that precedes the period, see the *vsamhlq* variable in Table 19 on page 55. The name of the file, following the period, can change; the MVS programmer who creates the VSAM data sets usually decides these names.	`'pkisrvd.vsam.icl.sceptid'` Note that this begins with the VSAM high-level qualifier.
`RemoveCompletedReqs`	Time period that completed certificate requests remain in the object store before automatic deletion.This is a number followed by `d` (days) or `w` (weeks). If not specified, the default is `1w` (1 week). The value `0d` disables the deletion of completed requests (not suggested).	UNIX programmer decides this value.	`1w`
`RemoveInactiveReqs`	Time period that incomplete, inactive certificate requests remain in the object store before automatic deletion. This is a number followed by `d` (days) or `w` (weeks). If not specified, the default is `4w` (4 weeks). The value `0d` disables the deletion of inactive requests (not suggested).	UNIX programmer decides this value.	`4w`
`RemoveExpiredCertsAndKeys`	Time period that keys and expired certificates with keys generated by PKI Services remain in the ICL and TKDS before automatic deletion. This is a number followed by `d` (days) or `w` (weeks). If you do not specify this parameter, or you set the value to `0d`, expired certificates are not removed.	UNIX programmer decides this value.	`520w`
`RemoveExpiredCerts`	Time period that expired certificates with keys that were not generated byPKI Services remain in the ICL before automatic deletion. This is a number followed by `d` (days) or `w` (weeks). If you do not specify this parameter, or you set the value to `0d`, expired certificates are not removed.	UNIX programmer decides this value.	`0d`
`SharedPLEX`	Indicates whether you intend to share a single copy of the PKI Services object store and the issued certificate list (ICL) among multiple images in a sysplex. This is T (True) or F (False). **Note:** This keyword has the same meaning as the `SharedVSAM` keyword in releases before z/OS V1R13. If the SharedVSAM parameter is present from an earlier release, it continues to work. If both SharedVSAM and SharedPLEX are present, SharedPLEX takes precedence.	UNIX programmer decides this value.	F
CertPolicy section			

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`AdminGranularControl`	Enables granular authority control for administrative functions that are based on CA domain name, certificate template name, and the administrative function being performed. If enabled, appropriate RACF protection profiles must be set up. If T (True), granular authority control is enabled. If F (False), granular authority is disabled. F is the default.	UNIX programmer decides this value.	F
`AdminNotifyNew`*n*	The email address to which notification should be sent immediately when a request is created and requires approval. The notification is only sent once. There can be multiple entries, where *n* is 1 for the first entry and increases sequentially for additional entries. The mailing address is in the form `<userid>@<system>`.	UNIX programmer decides this value. Do not change this information until you set up administrator notification of requests pending approval.	`abigail@`*mycompany*`.com`
`AdminNotifyReminder`*n*	The email address to which reminder notifications of requests pending approval should be sent when the daily maintenance task runs. There can be multiple entries, where *n* is 1 for the first entry and increases sequentially for additional entries. The mailing address is in the form `<userid>@<system>`.	UNIX programmer decides this value. Do not change this information until you set up administrator notification of requests pending approval.	`abigail@`*mycompany*`.com`
`ARLDist`	Indicates whether an authority revocation list (ARL) distribution point is created. F (the default) indicates that no ARL distribution point is created. T indicates that an ARL distribution point is created if `CRLDistSize` is greater than zero.	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See "Creating a distribution point ARL" on page 295 for more information.	F
`CertValidityConstraint`	Specifies whether the validity period of a certificate should be constrained within the CA's certificate life time. If T (True), requests with a validity period that exceeds the CA's validity period fail. If F (False), requests are not constrained to the CA's validity period. F is the default.	UNIX programmer decides this value.	F

`CPS`*n*	The Uniform Resource Identifier (URI) for the Certification Practice Statement (CPS) that is associated with `PolicyName`*n*. The value is in the form:
	`http://www.`*mycompany*
	`.com/cps.html`
		Do not change this information until you perform advanced customization. See "Using certificate policies" on page 284 for more information.	`http://`
			`www.`*mycompany*`.com/`
			`cps.html`
			If you changed `PolicyRequired=F` to `PolicyRequired=T`, you need to replace the sample value with a valid URI to your published Certificate Practice Statement.

`CreateInterval`	How often the certificate creation thread scans the database for approved requests. This is a number followed by `w` (weeks), `d` (days), `h` (hours), `m` (minutes), or `s` (seconds).	UNIX programmer decides this value.	3m

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`CRLDistDirPath`	The full path for the file system directory where PKI Services is to save each DP CRL, as specified by the HTTP URI in the CRLDistributionPoints extension. This value is ignored if you do not create a CRLDistributionPoints extension or if the URI protocol is `ldap`. This value can be specified with or without the trailing slash. The default value is `/var/pkiserv/`.	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See "Customizing distribution point CRLs" on page 290 for more information.	`/var/pkiserv/`
`CRLDistName`	Constant portion of the (leaf-node) relative distinguished name for a distribution point (DP) CRL, if DP CRL processing is being performed. The default value is `CRL`.	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See "Customizing distribution point CRLs" on page 290 for more information.	`CRL`
`CRLDistSize`	An integer value that represents the maximum number of certificates that can appear on one DP CRL. If you do not specify this parameter, or you set the value to `0`, DP CRLs are not created.	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See "Customizing distribution point CRLs" on page 290 for more information.	`500`
`CRLDistURI`*n*	*Optional:* Specifies a URI format name for the DP CRL. You can specify multiple names using parameters `CRLDistURI1`, `CRLDistURI2`, and so forth. This value is ignored if you do not create DP CRLs by specifying `CRLDistSize` with a value greater than zero. Specify this only if you want a URI-format name, in addition to the distinguished name format, which is built in the CRLDistributionPoints extension.	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See "Customizing distribution point CRLs" on page 290 for more information.	-
`CRLDuration`	The amount of time that a certificate revocation list is valid. This is a number followed by `w` (weeks), `d` (days), `h` (hours), `m` (minutes), or `s` (seconds).	UNIX programmer decides this value.	2d
`CRLEnhancements`	This keyword was introduced in z/OS Version 2 Release 1 and Version 2 Release 2 with the PTFs for the new function APAR OA51588. In z/OS Version 2 Release 3, the CRL processing enhancements are used by default and can no longer be disabled. As a result, the `CRLEnhancements` keyword is no longer supported. In addition, the restriction of the `CRLDuration` value being rounded up to the nearest number of days is removed. The value that is specified for `CRLDuration` is used unmodified. For more information about unsupported keywords, see the notes section following this table.	N/A	

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`CRLIDPExt`	Specifies whether certificate revocation lists (CRLs) should be created with a critical Issuing Distribution Point (IDP) extension. If T (True), CRLs are created with a critical IDP extension. If F (False), CRLs are created without the IDP extension. The default is T.	UNIX programmer decides this value.	T
`CRLWTONotification`	Specifies whether a console message is issued when CRL processing ends. If set to none, no console message is issued. If set to `file`, a console message is issued after the CRL is available in the file system. This keyword is ignored if either of the following conditions are true: • HTTP protocol is not specified for CRL distribution. • Large CRL posting is not enabled.	UNIX programmer decides this value.	none
`EnableCMP`	Specifies whether support for certificate management protocol (CMP) messages is enabled. If T (True), CMP messages that are supported are accepted. If F (False), all CMP messages are rejected. F is the default.	UNIX programmer decides this value.	F
`EnableLargeCRLPosting`	Specifies whether large CRL posting is enabled. If T, CRLs are saved in a z/OS UNIX directory before the LDAP posting thread processes them. If F, CRLs are saved in the object store (either VSAM data set or Db2 table, depending on which you are using), and are subject to a size limit of approximately 32 KB. The default is F.	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See "Enabling support for large CRLs" on page 297 for more information.	F
`EnablePathLenConstraint`	Specifies whether certificate path length constraints are enforced by the CA. The value is T (True) or F (False). If T, the CA certificate is examined at initialization to verify that it meets path length constraint requirements. If so, the pathLenConstraint field is set in the basic constraints extension of the intermediate CA certificates created by this CA. If not specified, or F, certificate path length constraint is not enforced in the CA certificate used by the CA, and intermediate CA certificates created by this CA do not include a pathLenConstraint field in the basic constraints extension.	UNIX programmer decides this value.	F
`EnableEST`	Specifies whether Enrollment over Secure Transport protocol (EST) is allowed. This is T (True) or F (False).	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See Chapter 16, "Using Enrollment over Secure Transport (EST)," on page 341 for more information.	F

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`EnableSCEP`	Specifies whether Simple Certificate Enrollment Protocol (SCEP) is allowed. This is T (True) or F (False).	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See Chapter 15, “Enabling Simple Certificate Enrollment Protocol (SCEP),” on page 333 for more information.	F
`ESTCAFile`	Specifies the full path name of the file containing the DER-encoded EST certificate authority (CA) certificate. Required when EnableEST is set to T.	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See Chapter 16, “Using Enrollment over Secure Transport (EST),” on page 341 for more information.	`/var/pkiserv/estcacert.der`
`ESTTemplate`	Specifies the template nickname in the pkiserv.tmpl or pkitmpl.xml file that is to be used when preregistering EST clients and issuing certificates to EST client. This value is limited to 8 characters and is ignored if the value exceeds that length. Required when EnableEST is set to T.	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See Chapter 16, “Using Enrollment over Secure Transport (EST),” on page 341 for more information.	2YESTP
`ExpireWarningTime`	**Note:** You need a value for this parameter only if you are sending email notifications to users when certificates are expiring, or automatically renewing certificates when they are expiring and sending them to the owners. This parameter indicates how soon before certificate expiration to send a warning message or a renewed certificate (that is, the number of days or weeks before the day and time the certificate expires). If automatic certificate renewal is active, this parameter indicates how soon before certificate expiration to renew the certificate and send it to the owner. This name-value pair is optional. Its absence indicates that no expiration checking is performed and no automatic certificate renewal occurs. Also, if the name-value pair is present but has an incorrect value or if PKI Services is configured to operate without LDAP, no expiration checking or automatic certificate renewal is done.	UNIX programmer decides this value.	`4w`

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`LargeCRLPostPath`	The full path for the file system directory where PKI Services is to save each CRL for posting to LDAP, if support for large CRLs is enabled. This value can be specified with or without the trailing slash, and can be the same as the value of `CRLDistDirPath`. The default value is `/var/pkiserv/`.	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See "Enabling support for large CRLs" on page 297 for more information.	`/var/pkiserv/crls`
`MaxSuspendDuration`	The length of the certificate suspension grace period in weeks or days. This is a number followed by `w` (weeks) or `d` (days). Certificates that remain suspended for longer than this period are automatically revoked. If you do not specify this parameter, or you set it to `0d`, the grace period is unlimited.	UNIX programmer decides this value.	`120d`
`OCSPType`	The type of OCSP responder support wanted: • `none` (the default) • `basic` If you do not specify this parameter, or you set the value to `none`, the responder is not enabled.	Change to `basic` if you want to enable the responder.	`none`
`PathLength`	Specifies the certificate path length constraint value to be included in the basic constraints extension of intermediate CA certificates that are created by the CA. Valid values are 0 - 16. The value that is specified must be less than the pathLenConstraint value in the PKI CA certificate, if it is present. This keyword is ignored if the EnablePathLenConstraint keyword is not set to T.	UNIX programmer decides this value.	`1`
`PKCS12Content`	Specifies which certificates PKI Services includes in the PKCS#12 returned to the requester when the certificate is retrieved. (This only applies to certificates where PKI Services generated the public/private key pair). Acceptable values for this keyword are: • **I** - The returned PKCS#12 contains the requested certificate and private key and contains the CA certificate that is used to sign the requested certificate. This is the default value if the keyword is omitted or if an unsupported value is specified. • **C** - The returned PKCS#12 contains the requested certificate and private key. It also contains the complete signing certificate chain including the root CA certificate, provided the CA certificates are connected to the key ring. • **E** - The returned PKCS#12 contains the requested certificate and private key. None of the CA certificates of the signing chain are included.	UNIX programmer decides this value.	`I`

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`PolicyCritical`	Indicates whether the CertificatePolicies extension should be marked critical. The value is T (True) or F (False).	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See "Using certificate policies" on page 284 for more information.	F
`PolicyRequired`	Indicates whether the CertificatePolicies extension should be included in all certificates that are created. The value is T (True) or F (False). T indicates that the CertificatePolicies extension is added to all certificates, and includes all `PolicyName`*n* entries that are specified in the configuration file. Any policies that are specified in the `CertPolicies` input parameter or listed in the CONSTANT subsection in the template file are ignored. F indicates that the CertificatePolicies extension is added to a certificate only when a certificate policy is specified in the CertPolicies input parameter or in the CONSTANT section of the template when a certificate is requested.	UNIX programmer decides this value. Do not change this information until you perform advanced customization. See "Using certificate policies" on page 284 for more information.	F
`PolicyName`*n*	A list of CertificatePolicies extensions that are added to all created certificates when `PolicyRequired`=T. The policy name is the symbolic name for a certificate policy OID and must match the name of a policy that is listed in the **OIDs** section.	Do not change this information until you perform advanced customization. See "Using certificate policies" on page 284 for more information.	*MyPolicy* If you changed `PolicyRequired`=F to `PolicyRequired`=T, replace the name *MyPolicy* with the same policy name used in the **OIDs** section.
`Policy`*n*`Org`	The name of the organization that prepared the User Notice Reference information that is associated with `PolicyName`*n*. For example: `International Business Machines, Inc.`	Do not change this information until you perform advanced customization. See "Using certificate policies" on page 284 for more information.	`My Company, Inc.` If you changed `PolicyRequired`=F to `PolicyRequired`=T, you need to specify your own value for this.
`Policy`*n*`Notice`*m*	Specifies the number of a textual statement, which is prepared by `Policy`*n*`Org` for the User Notice Reference that is associated with `PolicyName`*n*. More than one textual statement can apply.	Do not change this information until you perform advanced customization. See "Using certificate policies" on page 284 for more information.	`1` If you changed `PolicyRequired`=F to `PolicyRequired`=T, you need to specify your own value for this parameter.

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`SigAlg1`	The nickname that is assigned to the Object ID for the signature algorithm in the **OIDs** section. The supported algorithms and their nicknames are listed in Table 49 on page 287.	The supported algorithms and their nicknames are listed in Table 49 on page 287. Do not change this information until you perform advanced customization. See “Updating the signature algorithm” on page 287 for more information.	`sha-256WithRSAEncryption`
`TimeBetweenCRLs`	How often a certificate revocation list (CRL) should be created. This is a number followed by `w` (weeks), `d` (days), `h` (hours), `m` (minutes), or `s` (seconds). **Tip:** If you want PKI Services to create a CRL immediately, instead of waiting for the `TimeBetweenCRLs` interval to pass, use the `createcrls` utility. For more information, see “Using the createcrls utility” on page 431.	UNIX programmer decides this value.	`1d`
`UserNoticeText`*n*	The User Notice Explicit Text information that is associated with `PolicyName`*n*For example: `Certificate for IBM internal use only`. For the CA to conform with current standards, this textual statement must not exceed 200 characters.	Do not change this information until you perform advanced customization. See “Using certificate policies” on page 284 for more information.	*statement* If you changed PolicyRequired=F to PolicyRequired=T, you need to replace the variable *statement* with your own value.
General section			
`ExitTimeout`	Length of time that PKI Services waits for the **autorenew** preprocessing and postprocessing exit to return. If not specified, PKI Services waits for at most 30 seconds. PKI Services cancels the exit program if it runs longer than the specified time. The maximum value that is allowed is 1 hour. Any time specified greater is run for the maximum amount of time.	UNIX programmer decides this value.	`10s`
`InitialThreadCount`	Number of threads (at least 2 and no more than 100) the PKI Services daemon should create at program initialization.	UNIX programmer decides this value.	`10`
`MaintRunDays`	The days on which the daily maintenance task is to run. This is a list of digits between 0 and 6, representing the days of the week, with 0 representing Sunday, and 6 representing Saturday. The digits that are listed represent the days on which the task is to run. No spaces or other characters can be specified, and digits cannot be repeated. The digits can be specified in any order. If not specified, the task runs every day.	UNIX programmer decides this value.	`0123456`

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`MaintRunTime`	The time (local time) at which the daily maintenance task is to run, in the format *hh:mm*, where *hh* represents the hour (00 to 23) and *mm* represents the minutes (00 to 59). `00:00` represents midnight. If not specified, the task runs once per day at midnight local time.	UNIX programmer decides this value.	`01:00`
`RunMaintAtStart`	Indicates whether the daily maintenance task should run during PKI Services startup, in addition to the time and days that are specified by the `MaintRunTime` and `MaintRunDays` parameters. The value T (True) indicates that the task should run during PKI Services startup. The value F (False) indicates that the task should not run during PKI Services startup. If not specified, the daily maintenance task runs during PKI Services startup.	UNIX programmer decides this value.	`T`
`ReadyMessageForm`	The full path name or data set name containing the "Your certificate is ready" message form. • If you are not setting up PKI Services to generate keys for certificates, this name-value pair is optional. If you do not specify this name-value pair, no message is sent. • If you are setting up PKI Services to generate keys for certificates, this name-value pair is required. If you do not specify this name-value pair, requests to have PKI Services generate keys for certificates fail. **Guideline:** If you are adding a new CA domain, use the *ca_domain* value from Table 19 on page 55 as the second qualifier in the path name. **Example:** `/etc/pkiserv/employees/readymsg.form`	UNIX programmer decides this value.	`/etc/pkiserv/readymsg.form`
`RejectMessageForm`	The full path name or data set name containing the "Your certificate request has been rejected" message form. By default, no message is issued. Using this name-value pair is optional. **Guideline:** If you are adding a new CA domain, use the *ca_domain* value from Table 19 on page 55 as the second qualifier in the path name. **Example:** `/etc/pkiserv/employees/rejectmsg.form`	UNIX programmer decides this value.	`/etc/pkiserv/rejectmsg.form`
`ExpiringMessageForm`	The full path name or data set name containing the "Your certificate is about to expire" message form. By default, no message is issued. If your team specified a value for ExpireWarningTime (see the `ExpireWarningTime` row in this table), then ExpiringMessageForm is required. Otherwise, an error is logged and no expiring message processing is performed. **Guideline:** If you are adding a new CA domain, use the *ca_domain* value from Table 19 on page 55 as the second qualifier in the path name. **Example:** `/etc/pkiserv/employees/expiringmsg.form`	UNIX programmer decides this value.	`/etc/pkiserv/expiringmsg.form`

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`AdminNotifyForm`	The full path name or data set name containing the "request(s) pending for approval" message form. Defaults to no notification sent. **Guideline:** If you are adding a new CA domain, use the *ca_domain* value from Table 19 on page 55 as the second qualifier in the path name. **Example:** `/etc/pkiserv/employees/pendingmsg.form`	UNIX programmer decides this value.	**/etc/pkiserv/pendingmsg.form**
`AdminNotifyModForm`	The full path name or data set name containing the "request(s) approved with modifications" message form. Defaults to no notification sent.	UNIX programmer decides this value.	**/etc/pkiserv/pendingmsg2.form**
`RenewCertForm`	The full path name or data set name containing the "renewed certificate". Defaults to no certificate sent. **Guideline:** If you are adding a new CA domain, use the *ca_domain* value from Table 19 on page 55 as the second qualifier in the path name. **Example:** `/etc/pkiserv/employees/renewcertmsg.form`	UNIX programmer decides this value.	**/etc/pkiserv/renewcertmsg.form**
`RecoverForm`	The full path name or data set name containing the "list of certificates that satisfy your search criteria for recovery" message form. Use this name-value pair if you are setting up PKI Services to generate keys for certificate requests, and want users to be able to recover those certificates. **Guideline:** If you are adding a new CA domain, use the *ca_domain* value from Table 19 on page 55 as the second qualifier in the path name. **Example:** `/etc/pkiserv/employees/recoverymsg.form`	UNIX programmer decides this value.	**/etc/pkiserv/recoverymsg.form**
SAF section			
`KeyRing`	The fully qualified name of the SAF key ring for PKI Services to use. (This must consist of an uppercase user ID and a case-sensitive ring name that is separated by a slash (/.)	See the *ca_ring* and *daemon* values in Table 19 on page 55.	`PKISRVD/CAring`
`RA_label`	The label of your PKI Services registration authority (RA) certificate.	See the *ra_label* value in Table 11 on page 41.	`Local PKI RA`
`SecureKey`	Indicates whether keys generated by PKI Services are secure keys or clear keys. The value can be T (True) or F (False). T indicates that secure keys are generated in the TKDS. F or the absence of this keyword indicates that clear keys or secure keys are generated in the TKDS according to the installation configuration policy. `SecureKey` is ignored if `TokenName` is not specified.	UNIX programmer decides this value.	`F`

Table 21. Information needed for updating the configuration file (continued)

Parameter	Information needed	Where to get this information	Sample value or your customized value
`TokenName`	The name of a token in the ICSF PKCS #11 token data set (TKDS) that PKI Services uses to store key pairs that it generates for certificates. If this keyword is not specified, PKI Services cannot generate key pairs for certificates. If this keyword is specified, the TKDS must be set up before PKI Services starts. For information about setting up the TKDS, see *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*.	UNIX programmer decides this value. It must meet the requirements for a token name: • Up to 32 characters in length • Permitted characters are: – Alphanumeric – National: @ X'5B', # X'7B', or $ X'7C' – Period . X'4B' • The first character must be alphabetic or national • Lowercase letters can be used, but are folded to uppercase • The IBM1047 code page is assumed	`PKISRVD.PKIToken`
LDAP section			
	For information about the **LDAP** section, see Table 25 on page 108.		

Notes:

1. Keep in mind that everything in the `pkiserv.conf` file, including section names, keys, and values, is case-sensitive.
2. For boolean values, any of the following values are accepted for True: `T`, `t`, `Y`, `y`, or `1`. Any of the following values are accepted for False: `F`, `f`, `N`, `n`, or `0`
3. If an unsupported keyword is entered, or a supported keyword is entered in the wrong section of the `pkiserv.conf` file, the informational message IKYC045I is logged in the PKI Services job log. The keyword name and section are specified in the informational message. If this occurs, the keyword is ignored and PKI Services initialization continues.

Procedure

Perform the following steps to update the `pkiserv.conf` configuration file:

1. If necessary, update the **ObjectStore** section:

   a. If you want to use Db2 tables for the object store and issued certificate list (ICL) instead of VSAM files, uncomment the following line by removing the "# " at the beginning of the line:

   ```
   # DBType=DB2
   ```

   and leave the following line commented out:

   ```
   # DBType=VSAM
   ```

   If you want to use VSAM files, leave both lines commented out, or uncomment the following line:

```
# DBType=VSAM
```

b. If you are installing PKI Services for the first time, remove the # from the beginning of the line:

```
# DBVersion=1
```

If you are installing PKI Services from an earlier release and you want to use the SCEP requester enhancements available in Version 2 Release 3, you must upgrade all instances of PKI Services within the sysplex to the current release and convert the existing object store and ICL to the new format version. After converting the object store and ICL, remove the # from the beginning of the line:

```
# DBVersion=1
```

c. If `DBType` is set to `DB2` (see step "1.a" on page 88), uncomment the following lines, and if necessary change the Db2 package name and subsystem name in the following lines to the names you chose in the `DBPackage` and `DBSubsystem` rows in Table 21 on page 74:

```
# DBPackage=MasterCA

# DBSubsystem=DSN9


# DBWaitTime=30m
```

d. If `DBType` is set to `VSAM`, or commented out, or not present (see step "1.a" on page 88), change the data set names in the following lines, if necessary, to the names you chose in the `ObjectDSN`, `ObjectTidDSN`, `ObjectStatusDSN`, `ObjectRequestorDSN`, `ObjectSCEPTiDSN`, `ICLDSN`, `ICLStatusDSN`, and `ICLRequestorDSN` and `ICLSCEPTidDSN` rows in Table 21 on page 74. If `DBVersion` is set to `0`, commented out, or not present, do not uncomment or activate the `ObjectSCEPTidDSN` and `ICLSCEPTidDSN` entries.

For example, when `DBVersion` is set to a value of `0`:

```
DBVersion=0
ObjectDSN='pkisrvd.vsam.ost'
ObjectTidDSN='pkisrvd.vsam.ost.path'
ObjectStatusDSN='pkisrvd.vsam.ost.status'
ObjectRequestorDSN='pkisrvd.vsam.ost.requestr
# ObjectSCEPTidDSN='pkiservd.vsam.ost.sceptid'
ICLDSN='pkisrvd.vsam.icl'
ICLStatusDSN='pkisrvd.vsam.icl.status'
ICLRequestorDSN='pkisrvd.vsam.icl.requestr'
# ICLSCEPTidDSN='pkiservd.vsam.icl.sceptid'
```

When `DBVersion` is set to a value of 1:

```
DBVersion=1
ObjectDSN='pkisrvd.vsam.ost'
ObjectTidDSN='pkisrvd.vsam.ost.path'
ObjectStatusDSN='pkisrvd.vsam.ost.status'
ObjectRequestorDSN='pkisrvd.vsam.ost.requestr
ObjectSCEPTidDSN='pkiservd.vsam.ost.sceptid'
ICLDSN='pkisrvd.vsam.icl'
ICLStatusDSN='pkisrvd.vsam.icl.status'
ICLRequestorDSN='pkisrvd.vsam.icl.requestr'
ICLSCEPTidDSN='pkiservd.vsam.icl.sceptid'
```

If you are configuring PKI Services for the first time, be aware that the high-level qualifier of the VSAM data set names must match the name of the RACF user ID assigned to the PKI Services daemon (by default, PKISRVD). If you change from the default to another user ID, you need to change the high-level qualifier in the `pkiserv.conf` configuration file too. If the MVS programmer changes the data set names (see Step "2.d" on page 119), you must make equivalent changes in `pkiserv.conf`.

e. If necessary, change `1w` in the following line to the value in the `RemoveCompletedReqs` row in Table 21 on page 74:

```
RemoveCompletedReqs=1w
```

f. If necessary, change `4w` in the following line to the value in the `RemoveInactiveReqs` row in Table 21 on page 74:

```
RemoveInactiveReqs=4w
```

g. If necessary, uncomment the following line and, optionally, change `26w` to the value in the `RemoveExpiredCerts` row in Table 21 on page 74:

```
RemoveExpiredCerts=26w
```

h. If necessary, uncomment the following line and, optionally, change `520w` to the value in the `RemoveExpiredCertsAndKeys` row in Table 21 on page 74:

```
RemoveExpiredCertsAndKeys=520w
```

i. If necessary, update the SharedPLEX line:

- If you intend to use a sysplex and you are configuring PKI Services for the first time, change F in the following line to T:

  ```
  SharedPLEX=F
  ```

- If you are not using a sysplex (regardless of whether you are configuring PKI Services for the first time), you do not need to do anything.

____

2. If necessary, update the **CertPolicy** section.

   a. If necessary, change `3m` in the following line to the value in the `CreateInterval` row in Table 21 on page 74:

   ```
   CreateInterval=3m
   ```

   b. If necessary, update the `ExpireWarningTime` line or lines:

   - If you are sending email notifications and you are configuring PKI Services for the first time, if necessary change the value `4w` in the following line to the value in the `ExpireWarningTime` row of Table 21 on page 74.

     ```
     ExpireWarningTime=4w
     ```

   - If you are not using email notifications and you are configuring PKI Services for the first time, remove the `ExpireWarningTime=4w` line from the `pkiserv.conf` file.

   c. If necessary, change `1d` in the following line to the value in the `TimeBetweenCRLs` row in Table 21 on page 74:

   ```
   TimeBetweenCRLs=1d
   ```

   d. If necessary, change `2d` in the following line to the value in the `CRLDuration` row in Table 21 on page 74:

   ```
   CRLDuration=2d
   ```

   e. If necessary, change `F` in the following line to the value in the `PolicyRequired` row in Table 21 on page 74:

   ```
   PolicyRequired=F
   ```

   For more information about this parameter, see "Using certificate policies" on page 284.

f. If necessary, change F in the following line to the value in the `PolicyCritical` row in Table 21 on page 74:

```
PolicyCritical=F
```

For more information about this parameter, see "Using certificate policies" on page 284.

g. If necessary, change `120d` in the following line to the value in the `MaxSuspendDuration` row in Table 21 on page 74:

```
MaxSuspendDuration=120d
```

h. If necessary, establish distribution point (DP) certificate revocation lists (CRLs) and a DP authority revocation list (ARL). Follow the procedure that is shown in "Steps for customizing distribution point CRLs" on page 293 to determine the values for Table 21 on page 74.

i. If you want to enable the OCSP responder, change `OCSPType=none` to `OCSPType=basic`.

j. If you want to allow certificate management protocol (CMP) clients to send requests to PKI Services, change `EnableCMP=F` to `EnableCMP=T`. For information about support for CMP, see Chapter 21, "Using the certificate management protocol (CMP) with PKI Services," on page 453.

k. If you want to enable support for CRLs larger than 32 KB, change F in the following row to T and set `LargeCRLPostPath` to the full path of the file system directory where PKI Services is to save CRLs for posting to LDAP.

```
EnableLargeCRLPosting=F
```

For more information, see "Enabling support for large CRLs" on page 297.

l. If you want certificate revocation lists (CRLs) to be created without the Issuing Distribution Point (IDP) extension, uncomment the following line by removing the "#" character and change T to F:

```
#IDPExtCRL=T
```

m. If you want to enable Enrollment Over Secure Transport (EST), change "EnableEST=D to EnableEST=T. Other configuration options also control EST function. See Chapter 16, "Using Enrollment over Secure Transport (EST)," on page 341.

n. If you want to enable Simple Certificate Enrollment Protocol (SCEP), change `EnableSCEP=F` to `EnableSCEP=T`. See Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333.

o. If you want to enable granular authority control for administration functions, uncomment the following line by removing the "#" character and change F to T:

```
#AdminGranularControl=F
```

Before you enable granular authority control, the security administrator must set up profiles in the PKISERV class to control which functions each PKI administrator can perform. For more information, see "Using the PKISERV class to control access to administrative functions" on page 499.

p. If you want to enable certificate path length constraint, uncomment the following line by removing the "#" character and change F to T:

```
#EnablePathLenConstraint=F
```

Then uncomment the following line by removing the "#" character and change 1 to the value that you want the pathLenConstraint field to be set to in the basic constraints extension of intermediate CA certificates that are created by the CA. The value that you specify must be in the range 0 - 16, and must be less than the value of pathLenConstraint in the PKI CA certificate if it is present.

```
#PathLength=1
```

____

3. If you want a console message to be issued when CRL processing finishes, change the following line

   ```
   CRLWTONotification=none
   ```

   to

   ```
   CRLWTONotification=file
   ```

____

4. If necessary, update the **General** section:

   a. If necessary, change `10` in the following line to the value in the `InitialThreadCount` row in Table 21 on page 74:

      ```
      InitialThreadCount=10
      ```

   b. If necessary, change `10` in the following line to the value in the `ExitTimeout` row in Table 21 on page 74:

      ```
      ExitTimeout=30s
      ```

   c. If you choose, you can customize the time at which the daily maintenance task runs, the days on which it runs, and whether it also runs when the PKI Services daemon starts. This task is named daily_Timer, and performs functions such as:

      - Removing old and expired certificates
      - Removing inactive and completed certificate requests from the object store
      - Updating the low CRL distribution point that is based on expired certificates
      - Processing certificate expiration notification warning messages and automatic certificate renewal messages

      To specify that the daily maintenance task is to run at a time other than the default of midnight local time, remove the "#" from the following line and change `01:00` to the time that you want the task to run.

      ```
      #MaintRunTime=01:00
      ```

      To specify the days on which the daily maintenance task is to run, remove the "#" from the following line and change `0123456` to the list of digits representing the days on which you want the task to run. For example, specify 15 to run the task every Monday and Friday.

      ```
      #MaintRunDays=0123456
      ```

      To specify that the daily maintenance task is not to run when the PKI Services daemon starts, remove the "#" from the following line and change T to F.

      ```
      #RunMaintAtStart=T
      ```

   d. If necessary update the ReadyMessageForm, RejectMessageForm, ExpiringMessageForm, AdminNotifyForm, RenewCertForm, and RecoverForm lines:

      - If you are sending email notifications and you are configuring PKI Services for the first time, if necessary, change the values of the path name in the following lines to the corresponding values in Table 21 on page 74:

        ```
        ReadyMessageForm=/etc/pkiserv/readymsg.form

        RejectMessageForm=/etc/pkiserv/rejectmsg.form

        ExpiringMessageForm=/etc/pkiserv/expiringmsg.form

        AdminNotifyForm=/etc/pkiserv/pendingmsg.form
        ```

```
AdminNotifyModForm=/etc/pkiserv/pendingmsg2.form

RenewCertForm=/etc/pkiserv/renewcertmsg.form
```

- If you are allowing PKI Services to generate key pairs for certificates, if necessary change the value of the path name in the following lines to the corresponding value in Table 21 on page 74:

```
ReadyMessageForm=/etc/pkiserv/readymsg.form

RecoverForm=/etc/pkiserv/recoverymsg.form
```

- If you are not sending email notifications and you are configuring PKI Services for the first time, comment out the following lines in the `pkiserv.conf` configuration file by putting a "#" character in the first position of each line that does not already have a "#" character in it, as shown here:

```
# full pathname or data set name containing the 'your certificate
request
# has been rejected' message form. Defaults to no message
issued
# RejectMessageForm=/etc/pkiserv/
rejectmsg.form

# full pathname or data set name containing the 'your certificate is
about
# to expire' message form. Defaults to no message
issued
# ExpiringMessageForm=/etc/pkiserv/
expiringmsg.form

# full pathname or data set name containing the ‘request(s) pending
for
# approval’ message form. Defaults to no notification
sent
# AdminNotifyForm=/etc/pkiserv/
pendingmsg.form

# full pathname or data set name containing the request(s) approved
# with modifications message form. Defaults to no notification sent.
# AdminNotifyModForm=/etc/pkiserv/pendingmsg2.form

# full pathname or data set name containing the renewed
certificate.
# Defaults to no certificate
sent
# RenewCertForm=/etc/pkiserv/
renewcertmsg.form
```

- If you are not sending email notifications and you are not allowing PKI Services to generate keys for certificates, and you are configuring PKI Services for the first time, comment out the following lines in the `pkiserv.conf` configuration file, as shown here:

```
# full pathname or data set name containing the 'your certificate is
ready'
# message form. Defaults to no message
issued
# ReadyMessageForm=/etc/pkiserv/
readymsg.form
```

- If you are not allowing PKI Services to generate keys for certificates, and you are configuring PKI Services for the first time, comment out the following lines in the `pkiserv.conf` configuration file, as shown here:

  ```
  # full pathname or data set name containing information on certificate(s)
  # needed to be recovered.
  # RecoverForm=/etc/pkiserv/recoverymsg.form
  ```

- If you are allowing PKI Services to generate keys for certificates, and you are not configuring PKI Services for the first time, and you previously deleted the following lines in the `pkiserv.conf` configuration file, restore the following lines:

  ```
  # full pathname or data set name containing the 'your certificate is
  ready'
  # message form. Defaults to no message
  issued
  ReadyMessageForm=/etc/pkiserv/
  readymsg.form
  ```

____

5. If necessary, update the **SAF** section:

   a. If necessary, change `PKISRVD/CAring` in the following line to the value in the `KeyRing` row in Table 21 on page 74:

   ```
   KeyRing=PKISRVD/CAring
   ```

   b. If you specified `EnableSCEP=T` in Step “2.n” on page 91, change `Local PKI RA` in the following line to the value in the `ra_label` row in Table 11 on page 41:

   ```
   RALabel=Local PKI RA
   ```

   c. If you want PKI Services to be able to generate and store key pairs for certificate requests, and the ICSF programmer set up the ICSF PKCS #11 token data set (TKDS), uncomment the following line and change `PKISRVD.PKIToken` to the value you chose in the `TokenName` row in Table 21 on page 74:

   ```
   TokenName=PKISRVD.PKIToken
   ```

____

6. Restart PKI Services. Your changes do not take effect until you do this. For information about starting PKI Services see Chapter 10, “Starting and stopping PKI Services,” on page 129.

____

Updating pkiserv.conf after installing a new release of z/OS

After you install a new release of z/OS, you have two versions of the `pkiserv.conf` file on your system:

- The version that you used on the previous release of z/OS, which contains the changes you made to configure PKI Services.
- The sample version shipped with the new release of z/OS, which contains changes that IBM made to support the new function shipped in the release. This version does not contain any of your configuration changes.

You can continue to use the version from the previous release, but you might not be able to use the new function shipped in the new release until you merge the configuration changes you made in your version of the file with the updates that IBM made in the version shipped in the new release. One approach to merging the files is to compare your version of `pkiserv.conf` with the new sample version, located by default in the `/usr/lpp/pkiserv/samples/` directory. Update your version to match the changes made to the sample version. You should be able to cut and paste between the two versions of the file. To identify the lines that changed, you can refer to the sample of `pkiserv.conf` in Chapter 28, “The pkiserv.conf configuration file,” on page 607, which marks the lines that were changed with a bar in the left margin. But be aware that the sample shown might not be identical to the sample shipped with PKI Services.

Steps for setting up the var directory

You need to perform this task only if you are configuring PKI Services for the first time or adding a new CA domain.

Before you begin

Replace the following default values (used in the command examples) with values appropriate for your configuration:

Default value	Your value
PKISRVD	Use your daemon value in Table 19 on page 55.
`'pkisrvd.webroot.derbin'`	Use your `export_dsn` value in Table 19 on page 55.
`'pkisrvd.cacert.derbin'`	Use your `cacert_dsn` value in Table 19 on page 55.
`/var/pkiserv`	**Guideline:** Use your `ca_domain` value from Table 19 on page 55 to qualify the directory location if you are adding a new CA domain. For example, `/var/pkiserv/employees`.

Procedure

Perform the following steps to set up a UNIX directory and copy certain files that PKI Services needs into that directory:

1. Change ownership of the directory to the user ID of the PKI Services daemon by entering the following command from the UNIX command line:

   Example:

   ```
   chown PKISRVD /var/pkiserv
   ```

   ____

2. Copy the required certificates from their MVS data sets to the `/var/pkiserv` directory.

   Copy the web server certificate to the `cacert.der` file by entering the following command from the UNIX command line.

   Example:

   ```
   cp "//'pkisrvd.webroot.derbin'" /var/pkiserv/cacert.der
   ```

   If the Enroll over Secure Transport protocol (EST) is to be enabled for this certificate authority, copy the EST CA certificate to the `estcacert.der` file by entering the following on the UNIX command line:

   Example:

   ```
   cp "//'pkisrvd.cacert.derbin'" /var/pkiserv/estcacert.der
   ```

   ____

3. Copy the permission settings of the certificate files.

   Change the permissions for the Web Server certificate file by entering the following command from the UNIX command line.

Example:

```
chmod 644 /var/pkiserv/cacert.der
```

If an Enroll over Secure Transport protocol (EST) CA certificate file was created, change the permissions of this file by entering the following on the UNIX command line:

Example:

```
chmod 644 /var/pkiserv/estcacert.der
```

____

4. Change the ownership of the certificate files by entering the following command from the UNIX command line:

   Example:

   ```
   chown pkisrvd /var/pkiserv/*
   ```

____

Chapter 6. Tailoring the LDAP configuration for PKI Services

If you are configuring PKI Services for the first time, the LDAP programmer needs to load the LDAP schema file.

If you intend to use a non-z/OS LDAP product, refer to the documentation for that product. See Appendix A, "LDAP directory server requirements," on page 685 for information about installing a non-z/OS LDAP.

If you are configuring PKI Services for the first time, the LDAP programmer needs to set up an LDAP access control list (ACL) to allow any user to read CRLs, and might also need to set up another LDAP ACL to allow the distinguished name used for LDAP binding to create certificates and CRLs. For more information, see "Setting up authorization to create and access CRLs and certificates" on page 98.

You can optionally set up a secure connection with the LDAP server. For more information, see "Establishing a secure connection with LDAP (optional)" on page 98.

Steps for loading schema.user.ldif

Before you begin

- You need LDAP programming skills to complete this procedure.
- Make sure that the LDAP server is started before beginning these steps. If you are unsure about this, see "Steps for installing and configuring LDAP" on page 31.
- You need to know the following information from LDAP installation. Copy the information into the following table from (completed) Table 9 on page 32:

Table 22. LDAP information you need for tailoring LDAP configuration

LDAP information	**Explanation**	**Value**
Administrator's distinguished name	This is the distinguished name to use for LDAP binding. (For a definition of distinguished name, see Table 9 on page 32. The LDAP administrator defines the administrator's distinguished name with the **adminDN** keyword in the LDAP server configuration file. For example, the value is `"cn=Admin"` in `adminDN "cn=Admin"`	
Administrator password	This is the password to use for LDAP binding. The LDAP programmer can set this in several ways, for example: – By specifying the password as a TDBM entry by using the **userPassword** attribute in the **ldif2tdbm** load utility – By using the **adminPW** keyword in the LDAP server configuration file (not suggested)	
LDAP fully qualified domain name and port	This is the IP address and port on which the LDAP server is listening. For example, for `ldap.widgets.com:389`, the fully qualified domain name is `ldap.widgets.com` and the port is 389. See Table 8 on page 31 for a definition of fully qualified domain name. You can specify this address with or without the preceding string "ldap://" or "ldaps://".	

Table 22. LDAP information you need for tailoring LDAP configuration (continued)

LDAP information	Explanation	Value
Suffix	(For a definition of suffix, see Table 9 on page 32.) The suffix value is specified after the suffix keyword in the LDAP server configuration file. `suffix "o=your-company,c=your-country-abbreviation"`	

You need to load the `schema.user.ldif` file only if you are configuring PKI Services for the first time, whether you are using LDBM or TDBM. For more information, see the chapter on LDAP directory schema in *z/OS IBM Tivoli Directory Server Administration and Use for z/OS*.

Procedure

1. If you are configuring PKI Services for the first time, issue the following command to load the schema. Replace *adminDN*and *passwd*with the adminDN and adminPW values from Table 22 on page 97.

   ```
   ldapmodify -D adminDN -w passwd -f /usr/lpp/ldap/etc/schema.user.ldif
   ```

Setting up authorization to create and access CRLs and certificates

Certificate revocation lists (CRLs) in an LDAP directory have an attribute of **critical**, which allows only the LDAP administrator to read them. If you are configuring PKI Services for the first time, the LDAP programmer needs to set up an LDAP access control list (ACL) to allow users other than the LDAP administrator to read CRLs. If the ACL is not set up, only the LDAP administrator can retrieve CRLs from LDAP. Other users might get access violation messages if they attempt to retrieve a CRL from LDAP, and LDAP does not return the CRL.

In addition, if the distinguished name to be used for LDAP binding is not the LDAP administrator, the LDAP programmer needs to set up another LDAP ACL to allow that distinguished name to create CRLs and certificates. You define the distinguished name to be used for LDAP binding in the AuthName1 line of the `pkiserv.conf` file. For more information about the AuthName1 line, see Chapter 8, “Tailoring the PKI Services configuration file for LDAP,” on page 107.

For information about setting up LDAP ACLs, see the information about access control in *z/OS IBM Tivoli Directory Server Administration and Use for z/OS*.

Tips: When setting up an LDAP ACL for PKI Services, consider these facts:

- You can use the **entryOwner** attribute to allow an application to read and write LDAP entries without having to use the LDAP administrator bind credentials.
- You can use a propagating ACL (the **aclPropagate** attribute is set to **TRUE**) to allow the defined ACL to cover new CRLs created by PKI Services.

Establishing a secure connection with LDAP (optional)

You can optionally set up a secure connection between PKI Services and the LDAP server to prevent the bind password from flowing in the clear. The secure connection uses the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols, provided by z/OS Cryptographic Services System SSL services, to maintain an encrypted communications path between PKI Services and the LDAP server. For information about how to configure LDAP to use a secure connection, see the topic on using SSL/TLS protected communications in *z/OS IBM Tivoli Directory Server Administration and Use for z/OS*.

If you are using a secure connection with LDAP, the RACF administrator needs to add a certificate to the PKI Services key ring for validating the LDAP server:

- If the LDAP server you are using is using a self-signed certificate, add that self-signed certificate to the PKI Services key ring.
- If the LDAP server is using a certificate signed by a certificate authority (CA), add the certificate for the CA to the PKI Services key ring, if it is not already there. Use whatever means the CA provides to obtain the CA's certificate.

For the name of the PKI Services key ring, see Table 19 on page 55. The RACF administrator uses the RACF RACDCERT command to add a certificate to the key ring. For information about RACDCERT, see *z/OS Security Server RACF Command Language Reference*.

Chapter 7. Updating IBM HTTP Server - Powered by Apache configuration and starting the server

PKI Services uses the IBM HTTP Server - Powered by Apache for the following functions:

- The PKI Services web application, if you implement it using REXX CGI scripts.
- OCSP support
- SCEP support
- EST support
- CMP support

You need to perform the tasks in this topic only if you are configuring PKI Services for the first time and use one or more of these functions. If you do not use any of these functions, you can skip the tasks in this topic.

PKI Services supports IBM HTTP Server - Powered by Apache. It is part of the IBM z/OS operating system. For more information, see IBM HTTP Server (www.ibm.com/software/products/http-servers). It is also included with WebSphere Application Server.

Setting up IBM HTTP Server - Powered by Apache

Starting the web server requires having a configuration file for it. This topic describes how the web server programmer performs the following tasks for IBM HTTP Server httpd.conf:

- Updating the IBM HTTP Server - Powered by Apache configuration files by cutting and pasting directives from the PKI Services samples directory into them
- Starting the IBM HTTP Server - Powered by Apache.

Before you begin

- The IBM HTTP Server - Powered by Apache must already be configured.
- It would be helpful to have the documentation for IBM HTTP Server available, see the WebSphere Application Server Knowledge Center (www.ibm.com/support/knowledgecenter/SSEQTP).
- The keyring that would be specified in the SSL configuration files must exist and have the web server certificate and its CA chain up to the root connected to it.
- The CA root certificate must be distributed to the users who require access to the PKI web page interfaces. A possible method is to distribute the CA root certificate by using the same communication channel that you use to provide the users with the URI of the PKI web page. For example, if you send email, you can:
  1. Add the CA root certificate as an attachment or include it as Base64 encoded text.
  2. Send a separate communication with the root certificate fingerprint to help users ensure that the correct CA certificate was received in the previous email.

Steps for updating the IBM HTTP Server - Powered by Apache configuration files

PKI Services ships sample IBM HTTP Server - Powered by Apache configuration files

- The main configuration file (`httpd.conf`)
- Virtual host files:
  - `vhost80.conf` - Virtual Host file for non-SSL requests
  - `vhost443.conf` - Virtual Host file for SSL requests with server authentication

– `vhost1443.conf` - Virtual Host file for SSL requests with client authentication

These files are used by the IP-based virtual hosting feature of the IBM HTTP Server - Powered by Apache. IP-based virtual hosting is a method to apply different directives that are based on the IP address and port on which a request is received. PKI Services provides sample virtual host files for non-SSL requests, SSL requests, and SSL requests with client authentication on different ports.

Table 23 on page 102 summarizes the virtual host files that are used for normal HTTP traffic and SSL traffic with specific ports.

Table 23. Virtual host files

Virtual host configuration file	Protocol	SSL	Server authentication	Client authentication	Port number
vhost80.conf	HTTP	No	No	No	80
vhost443.conf	HTTPS	Yes	Yes	No	443
vhost1443.conf	HTTPS	Yes	Yes	Yes	1443

Before you begin

- You must perform these steps only if you are configuring IBM HTTP Server - Powered by Apache for PKI Services for the first time. If you are using IBM HTTP Server - Powered by Apache and you are now updating the configuration for use with PKI Services, some of the following steps are not required.
- This information assumes that you used the installer program (`bin/install_ihs`) to install IBM HTTP Server - Powered by Apache. You must know the installation directory for the server instance, which is referred to as *ihs-install-dir* in the sample commands. (This directory must be different from the product directory, usually `/usr/lpp/ihsa_zos`).
- You must know the file system installation directory (the file system directory where the MVS programmer installed PKI Services), called *pki-install-dir* in the commands that follow. The default is `/usr/lpp/pkiserv/`. The MVS programmer was asked to record any changes to the defaults; see Table 3 on page 11.
- You must know the following LDAP information. Record the information in the rightmost column of Table 24 on page 102.

  Note: The default name of the LDAP server configuration file is `ds.conf` for the LDAP server that is provided by IBM Tivoli Directory Server.

Table 24. LDAP information you need for tailoring IBM HTTP Server - Powered by Apache configuration

LDAP information	Explanation	Value
Administrator's distinguished name	The distinguished name to use for LDAP binding. (For a definition of distinguished name, see Table 9 on page 32.) The LDAP administrator defines the administrator's distinguished name with the adminDN keyword in the LDAP server configuration file. For example, the value is "`cn=Admin`" in `adminDN "cn=Admin"`	
Administrator password	The password to use for LDAP binding. The LDAP programmer can set this password in several ways; for example: – By specifying the password as a TDBM entry by using the `userPassword` attribute in the **ldif2tdbm** load utility – By using the adminPW keyword in the LDAP server configuration file (not suggested)	

Table 24. LDAP information you need for tailoring IBM HTTP Server - Powered by Apache configuration (continued)

LDAP information	Explanation	Value
LDAP fully qualified domain name	The IP address on which the LDAP server is listening, for example, `ldap.widgets.com`. See Table 8 on page 31 for a definition of fully qualified domain name.	
LDAP port	The port for LDAP, for example, 389 in `ldap.widgets.com:389`	

Procedure

Perform the following steps to update the IBM HTTP Server - Powered by Apache configuration files:

1. Copy the IBM HTTP Server - Powered by Apache directives from the PKI Services samples configuration file, *`pki-install-dir`*`/samples/httpd.conf` to the HTTP server configuration file, *`ihs-install-dir`*`/conf/httpd.conf`, and make the following updates.

   Note: The file *`pki-install-dir`*`/samples/httpd.conf` is not a complete `httpd.conf` file. It contains only the directives that might not be present in your `httpd.conf` file and that might be unique to the PKI Services CGI scripts and programs.

   a. Use the load module directive to add the required modules to the list of modules, if they do not exist:

      - `rewrite_module modules/mod_rewrite.so`
      - `authnz_saf_module modules/mod_authnz_saf.so`
      - `ibm_ssl_module modules/mod_ibm_ssl.so`
      - `alias_module modules/mod_alias.so`

   b. Add the `addtype` directives to your list of `addtypes` if they do not exist:

      ```
      AddType application/x-x509-user-cert .cer
      AddType application/x-x509-ca-cert .der
      AddType application/octet-stream .msi
      AddType application/pkix-crl .crl
      ```

   c. Copy the `Keyfile` and the `Include` directives as is, replacing any existing values.

   d. If your organization customized the value of `web_ring` (see Table 11 on page 41), change `SSLring` in the `Keyfile` directive in the following line to the customized value:

      ```
      Keyfile /saf SSLring
      ```

2. If the virtual host files (`vhost80.conf`, `vhost443.conf`, `vhost1443.conf`) do not exist, create them by copying them from *`pki-install-dir`*`/samples` to *`ihs-install-dir`*`/conf/vhost80.conf`, *`ihs-install-dir`*`/conf/vhost443.conf`, and *`ihs-install-dir`*`/conf/vhost1443.conf`. For example, assuming the default *`pki-install-dir`* and an *`ihs-install-dir`* of `/etc/websrv1`, the following command copies all three vhost files to the `/etc/websrv1/conf` directory:

   ```
   cp /usr/lpp/pkiserv/samples/vhost*.conf /etc/websrv1/conf
   ```

3. Make the following updates to each of the virtual host files:

   a. Change all instances of *server-domain-name* to the fully qualified domain name of your web server. For example, www.ibm.com. (For information about your web server's fully qualified domain name, see Table 8 on page 31.)

   b. Change all instances of *application-root* to the value of *pki-install-dir*, which is `usr/lpp/pkiserv` by default.

   c. If necessary, change the environment variable `_PKISERV_CONFIG_PATH` to identify the runtime directory of your CA domain. (See Table 53 on page 307.)

d. (Optional) If you intend to have a dedicated set of administrators for each CA domain, add an environment variable that specifies the runtime directory for each administrative domain. (See Table 53 on page 307.)

Example:

```
SetEnv _PKISERV_CONFIG_PATH_PKIServ "/etc/pkiserv"
```

Note: In the `vhost80.conf` file, which defines directives for non-SSL requests, a `Listen` directive is not specified. It is assumed that the `Listen` directive is defined in the main `httpd.conf` file that tells the server to accept incoming requests on the specified port. If you do not have a `Listen` directive in the `httpd.conf` file, add the `Listen 80` directive on the line before the `VirtualHost *:80` directive in the `vhost80.conf` file.

4. Perform the following step to update the `vhost443.conf` virtual host configuration file.

   a. If your organization customized the value of `web_ring` (see Table 11 on page 41), change `SSLring` in the `Keyfile` directive in the following line to the customized value:

   ```
   Keyfile /saf SSLring
   ```

5. Perform the following steps to update the `vhost1443.conf` virtual host configuration file.

   a. If your organization customized the value of `web_ring` (see Table 11 on page 41), change `SSLring` in the `Keyfile` directive in the following line to the customized value:

   ```
   Keyfile /saf SSLring
   ```

   b. If you would like the IBM HTTP Server to preform revocation checking, add the following directives after the `SSLClientAuth` directive:

   - `SSLCRLHostName`
   - `SSLCRLPort`
   - `SSLCRLUserID`
   - `SSLStashfile`

   Note: `SSLStashfile` is the fully qualified path to the file that contains the password for the user name on the LDAP server. This directive is not required for an anonymous bind. Use it when you specify a user ID. Use the **`sslstash`** command, which is in the `bin` directory of IBM HTTP Server, to create your CRL password stash file. Specify the password that you use to log in to your LDAP server as the password on the **`sslstash`** command. The format of the **`sslstash`** command is:

   ```
   sslstash [-c] file function password
   ```

   where:

   -c
   Creates a new stash file. If not specified, an existing file is updated.

   file
   Is the fully qualified name of the file to create or update.

   function
   Indicates the function for which the password is to be used. Valid values include `crl` and `crypto`.

   password
   Is the password to stash.

Starting and stopping the IBM HTTP Server - Powered by Apache

Steps for starting the IBM HTTP Server - Powered by Apache

Perform the following steps to start the IBM HTTP Server - Powered by Apache.

1. Make sure that the LDAP server is started. (For more information, see "Steps for installing and configuring LDAP" on page 31.)
2. Take one of the following actions:
   - Issue the following command from the IBM HTTP Server - Powered by Apache installation directory:

     ```
     ihs-install-dir/bin/apachectl start
     ```

   - To start the IBM HTTP Server - Powered by Apache using an alternative configuration file, issue the following command:

     ```
     apachectl -k start -f path_to_configuration_file
     ```

   - If you want to use the sample JCL procedure from `hlq.SIWOJCL(IWOAPROC)`, copy the JCL to the system procedure library. If you copy the JCL to a procedure called WEBSRV1, start the server by entering the following MVS console command:

     ```
     S WEBSRV1
     ```

Steps for stopping the IBM HTTP Server - Powered by Apache
To stop the IBM HTTP Server - Powered by Apache, perform the following step:

1. Enter the following MVS console command:

   ```
   S WEBSRV1,ACTION='stop'
   ```

Chapter 8. Tailoring the PKI Services configuration file for LDAP

You need to tailor the **LDAP** section of the `pkiserv.conf` configuration file only if you meet one of the following conditions:

- You are configuring PKI Services for the first time
- You intend to use encrypted passwords for your LDAP servers

Chapter 5, "Configuring the UNIX runtime environment," on page 67 describes tasks the UNIX programmer performs. The other team members perform additional tasks before the UNIX programmer updates the **LDAP** section of the `pkiserv.conf` configuration file (described in this topic) and starts the PKI Services daemon (described in Chapter 10, "Starting and stopping PKI Services," on page 129).

Excerpt of LDAP section

The following excerpt shows the **LDAP** section of the `pkiserv.conf` configuration file as it is shipped:

```
[LDAP]
NumServers=1
PostInterval=5m
Server1=myldapserver.mycompany.com:389
AuthName1=CN=root
AuthPwd1=root
CreateOUValue= Created by PKI Services
RetryMissingSuffix=T
# Name of the LDAPBIND Class profile containing the bind information for LDAP
# server 1. This key is optional. Used in place of keys Server1, AuthName1.
# and AuthPwd1
#BindProfile1=LOCALPKI.BINDINFO.LDAP1
```

You use the **LDAP** section of the `pkiserv.conf` file to provide information for one or more LDAP servers. The NumServers line specifies the number of servers.

Storing information for encrypted passwords for your LDAP servers

You store information about passwords for binding to LDAP directories in the `pkiserv.conf` configuration file. Passwords can be in clear text or encrypted. By default, the `pkiserv.conf` configuration file contains Server1, AuthName1, and AuthPwd1 parameters; these lines are for specifying your LDAP bind information, including passwords, in clear text: (For more than one LDAP server, you add additional lines, Server2, AuthName2, AuthPwd2, Server3, AuthName3, AuthPwd3, and so forth.) If you want to use encrypted passwords for your LDAP servers, you delete all these lines, uncomment (remove the #) from the BindProfile1 line at the end of the file, and correct the profile value that is specified, if necessary. (See "Using encrypted passwords for LDAP servers" on page 501 for information about setting up this bind profile in RACF). For more than one LDAP server, you add additional lines: BindProfile2, BindProfile3, and so forth.

PKI Services performs the following processing when locating LDAP bind information:

1. The Server*n* line specifies the fully qualified domain and port of your LDAP server. If your file contains a Server*n* line, PKI Services looks for the matching AuthName*n* and AuthPwd*n* lines and uses these values.
2. The BindProfile*n* parameter specifies the name of the LDAPBIND class profile. If your file does not contain a Server*n* line but does contain a BindProfile*n* line, PKI Services looks for the bind information

in the LDAPBIND class profile. (If Server*n* is present, PKI Services does not look for bind information in BindProfile*n*, even if the value in Server*n* is incorrect.)

3. If neither is present for a specific server, then PKI Services uses the default from IRR.PROXY.DEFAULTS in the FACILITY class.

Steps for tailoring the LDAP section of the configuration file

Before you begin

- **Important:** You need to update the **LDAP** section of the `pkiserv.conf` configuration file only if you are configuring PKI Services for the first time or your company is using encrypted passwords for your LDAP servers.
- You need UNIX programming skills to complete this procedure.
- Table 25 on page 108 lists some parameters that are in the **LDAP** section of the `pkiserv.conf` configuration file. The rightmost column lists the default values. You need to change some of these values. Fill in the blank lines with your company's information (and cross out these defaults). If you decide to change any of the other defaults, cross out these values and record your company's information.

Table 25. Information needed for updating the LDAP section of the configuration file

Parameter	Information needed	Where to get this information	Default value and your company's information
`NumServers=`	The number of available LDAP servers. These are replicas that can post certificates and CRLs.	From LDAP programmer	1

Table 25. Information needed for updating the LDAP section of the configuration file (continued)

Parameter	Information needed	Where to get this information	Default value and your company's information
`PostInterval=`	How often the posting thread scans the request database for certificates and CRLs to post to the LDAP server in weeks (`w`), days (d), hours (h), minutes (`m`), or seconds (s) if NumServers > 0. **Notes:** 1. If the post is unsuccessful for a certificate, the post is tried again at the next post interval. If the post continues to be unsuccessful after 3 attempts, the post frequency for this certificate is reduced to no more than once per hour. After 26 unsuccessful attempts, it is further reduced to no more than once per day. After 33 unsuccessful attempts, the post request for this certificate is deleted from the request database. 2. Certificates created when NumServers is set to 0 are not posted to LDAP. If the value of NumServers is changed later to enable posting, the new value applies to new certificates only.	UNIX programmer decides this. Specify a number followed by h (hours), `m` (minutes), or `s` (seconds). Example: `6m`	`5m`

Table 25. Information needed for updating the LDAP section of the configuration file (continued)

Parameter	Information needed	Where to get this information	Default value and your company's information
`Server1=`	You use this parameter only if you are storing LDAP passwords in the clear. This parameter's value is the fully qualified domain name (domain name or IP address and port) for the first LDAP server. If you are using a Secure Sockets Layer (SSL) session, the fully qualified domain name should be preceded by `ldaps://`.	Copy this information from the earlier (completed) table, Table 9 on page 32.	*myldapserver.mycompany*.com:389 ___________________________ **Note:** If the number of servers (the value in the row containing `NumServers=`) is greater than one, you need one value for *each* server.
`UseBinaryAttr1=`	Specifies whether the CA posts certificates and CRLs to the LDAP server with the binary attribute. Valid values are T (True) or F (False). If `NumServers` is greater than 1, specify a value for each server; for example, specify `UseBinaryAttr2` for server 2. If a value of UseBinaryAttr*n* is not specified, it defaults to F.	UNIX programmer decides this (after consulting with LDAP programmer)	F
`AuthName1=`	You use this parameter only if you are storing LDAP passwords in the clear. This parameter's value is the distinguished name to use for LDAP binding. (See Table 9 on page 32 for a definition of distinguished name.)	Copy this information from the earlier (completed) table, Table 9 on page 32.	*CN=root* **Notes:** • If the number of servers (the value in the row containing `NumServers=`) is greater than one, you need one value for *each* server. • The default name of the LDAP server configuration file is `ds.conf` for the IBM Tivoli Directory Server for z/OS LDAP server.

Table 25. Information needed for updating the LDAP section of the configuration file (continued)

Parameter	Information needed	Where to get this information	Default value and your company's information
`AuthPwd1=`	You use this parameter only if you are storing LDAP passwords in the clear. This parameter's value is the password to use for LDAP binding. The LDAP programmer sets this. **Note:** Include this parameter, `Server1`, and `AuthName1` only if you are storing the LDAP password in the clear. Alternately, if you encrypt the password for an LDAP server, use the `BindProfile1` parameter. Omitting `BindProfile1` and `Server1` specifies using the PROXY segment information from the IRR.PROXY.DEFAULTS profile in the FACILITY class. (For more information, see "Using encrypted passwords for LDAP servers" on page 501.)	Copy this information from the earlier (completed) table, Table 9 on page 32.	*root* ____________________ **Note:** If the number of servers (the value in the row containing `NumServers=`) is greater than one, you need one value for *each* server.
`CreateOUValue=`	Value to use for the OU attribute when creating LDAP entries under the objectclass organizationalUnit. (See Table 112 on page 685.) This is used only when no OU value is specified in the relative distinguished name.	UNIX programmer decides this (after consulting with LDAP programmer)	`Created by PKI Services`

Table 25. Information needed for updating the LDAP section of the configuration file (continued)

Parameter	Information needed	Where to get this information	Default value and your company's information
`RetryMissingSuffix=`	True (T) or False (F) setting that indicates whether LDAP post requests should be tried again later if the distinguished name suffix does not exist. When set to F, LDAP post requests that fail because of a missing suffix are discarded.	UNIX programmer decides this (after consulting with LDAP programmer)	T
`BindProfile1=`	You use this parameter only if you intend to use an encrypted password for your LDAP server. This parameter's value is the name of the LDAPBIND class profile containing the bind information for the LDAP server. (For more information, see "Using encrypted passwords for LDAP servers" on page 501.)	Get the profile name from the RACF administrator who creates the profile. See "Using encrypted passwords for LDAP servers" on page 501 for more information.	LOCALPKI.BINDINFO.LDAP1 _____________________ **Note:** If the number of servers (the value in the row containing `NumServers=`) is greater than one, you need one value for *each* server.

Procedure

Perform the following steps to update the **LDAP** section of the `pkiserv.conf` configuration file (if you are configuring PKI Services for the first time or using encrypted passwords for your LDAP servers):

1. If necessary, change 1 (the default) in the following line to the number of available LDAP servers listed in Table 25 on page 108:

   ```
   NumServers=1
   ```

   ____

2. Optionally change 5m in the following line to the posting interval in Table 25 on page 108:

   ```
   PostInterval=5m
   ```

   ____

3. If necessary, update the BindProfile1 line or the Server1, AuthName1, and AuthPwd1 lines:

   - If you intend to use an encrypted password for your LDAP server and you are configuring PKI Services for the first time, perform the following steps:

   a. If you are using an LDAPBIND class profile, remove the comment delimiter (#) from the start of the following line and change `LOCALPKI.BINDINFO.LDAP1` to the name of the LDAPBIND class profile. (See Step "3" on page 502).

   ```
   # BindProfile1=LOCALPKI.BINDINFO.LDAP1
   ```

   b. Delete the following three lines in the **LDAP** section:

   ```
   Server1=myldapserver.mycompany.com:389
   AuthName1=CN=root
   AuthPwd1=root
   ```

- If you are not using an encrypted password for your LDAP server and are configuring PKI Services for the first time, perform the following steps:

   a. Change *your-ldap-server-address:port* to your fully qualified domain name and port as listed in Table 25 on page 108:

   ```
   Server1=your-ldap-server-address:port
   ```

   b. Change *CN=root* in the following line to the value of the administrator distinguished name in Table 25 on page 108:

   ```
   AuthName1=CN=root
   ```

   c. Change *root* in the following line to the value of the administrator password in Table 25 on page 108:

   ```
   AuthPwd1=root
   ```

____

4. If the value of `NumServers=` is greater than 1, repeat Step "3" on page 112 for each additional server. (You need to increment the number in the parameter names for each additional server, for example `Server2`, `AuthName2`, `AuthPwd2`.

____

5. If necessary, change `Created by PKI Services` in the following line to the OU attribute value in Table 25 on page 108:

   ```
   CreateOUValue=Created by PKI Services
   ```

____

6. If necessary, change T in the following line to the RetryMissingSuffix value in Table 25 on page 108:

   ```
   RetryMissingSuffix=T
   ```

____

7. If you want certificates and CRLs posted to the LDAP server with the binary attribute, remove the comment delimiter (#) from the start of the following line and change F to T:

   ```
   #UseBinaryAttr1=F
   ```

   If the value of `NumServers=` is greater than 1, repeat this step for each server. Increment the number in the parameter name for each additional server, for example `UseBinaryAttr2`.

____

Chapter 9. Creating the object store and ICL

This topic includes the following procedures:

You need to perform the tasks in this topic if:

- You are configuring PKI Services for the first time or adding a new CA domain.
- Your organization is using a sysplex for PKI Services daemons.
- You want to tune VSAM performance.
- Your object store and ICL are in VSAM data sets, and you want to convert them to Db2 tables

The object store and ICL

PKI Services maintains two databases containing information about certificate requests and issued certificates:

- The *object store*, or *request database*, holds records to track active certificate requests and posting objects for certificates and certificate revocation lists. Object store records are not permanent. They are deleted when they are no longer needed:
  - CRL posting requests and certificate posting requests are removed when they are successfully posted to LDAP.
  - Revocation requests are deleted at the end of revocation processing.
  - When a certificate is retrieved by the requester, the certificate request is deleted after the time period specified by the parameter `RemovedCompletedReqs` in the configuration file (by default one week).
  - If a certificate is not retrieved by the requester, the certificate request is deleted after the time period specified by the parameter `RemovedInactiveReqs` in the configuration file (by default four weeks).
- The *issued certificate list (ICL)* contains a permanent record for each certificate that PKI Services issues. There is one ICL record for each issued certificate.

You have two options for implementation of the object store and ICL:

- VSAM data sets.
- Db2 tables. (This option was introduced in z/OS V1R13).

Some things to consider when choosing which option to use:

- VSAM is shipped with z/OS, but you must purchase Db2.
- VSAM supports limited query functions and does not allow users to create their own queries. Db2 allows users to create queries.

Certificate revocation lists (CRLs) created temporarily in the object store for LDAP posting are limited in size to approximately 32 KB. You can avoid this limitation by enabling support for large CRLs, which causes CRLs to be stored in the z/OS UNIX file system instead of the object store. For more information, see "Enabling support for large CRLs" on page 297.

Note: In Version 2 Release 3, the addition of new fields within the object store and ICL to support the SCEP requester enhancement cause the object store and ICL to be restructured. The old structure is considered to be a `version 0` format and the new structure is considered to be a `version 1` format. The following samples are added to create or alter the `version 1` object store and ICL.

- IKYCVSV1 (similar to IKYCVSAM)
- IKYCDBV1 (similar to IKYCDB2)
- IKYRVSV1 (similar to IKYRVSAM)

If you are configuring PKI Services for the first time or adding a new CA domain, use the new samples to create `version 1` object store and ICL. If the creation of the `version 0` object store and ICL is required, you can use the following original samples, IKYCVSAM, IKYCDB2, and IKYRVSAM.

Creating the object store and ICL using VSAM data sets

The MVS programmer performs the following tasks:

- If configuring PKI Services for the first time, create the VSAM object store and ICL data sets and indexes.
- If you want, tune VSAM data set performance.

Sysplex considerations

If you are configuring the PKI Services to run in a sysplex environment, as a single PKI instance, perform the preliminary steps for establishing VSAM record-level sharing or RLS.

If you want to enable the SCEP requester enhancement, ensure that all members are migrated to Version 2 Release 3. The migration to Version 2 Release 3 must be complete before the conversion of object store and ICL to the new version for each instance of PKI. If the PKI instance starts successfully from all of the members, you can start to convert the object store and ICL for each instance with the following steps:

- Run IKYCVSV1 to create the new VSAM data sets with the alternate index for the SCEP `transID` field.
- Stop the instance of PKI from all members.
- Run the conversion utility, `vsamconv`.
- Update `pkiserv.conf` to specify the `DBVersion` to 1 and point to the new VSAM data sets from all members.
- Start the instance of PKI from all members.

Planning VSAM storage requirements

The MVS programmer uses the IKYCVSV1 sample JCL to create two VSAM data sets (clusters):

- A data set for the request database (object store)
- A data set for the issued certificate list (ICL).

The MVS programmer also uses the same sample JCL to create five alternate index data sets (paths):

1. Transaction ID alternate index into the object store
2. Status alternate index into the object store
3. Requestor alternate index into the object store
4. Status alternate index into the ICL
5. Requestor alternate index into the ICL.

The IKYCVSV1 sample JCL contains default values for the primary and secondary extent allocations for these data sets. The default allocation for base clusters is `CYL(3,1)`. For alternate indexes, it is `TRK(5,1)`. You need to update these values based on your anticipated future needs. Use the guidelines in "Determining storage needs for the ICL" on page 117 and "Determining storage needs for the object store" on page 117 to update the space allocation parameters for the DEFINE CLUSTER and DEFINE ALTERNATEINDEX statements. (For more information about IDCAMS, see *z/OS DFSMS Access Method Services Commands*.)

Determining storage needs for the ICL

Unless set up otherwise, the ICL grows continuously over time as more certificates are issued. The size of a certificate varies depending on many factors; for example, the public key type and size, the length of the issuer and subject names, the number and size of Subject Alternative names, and the number and size of certificate extensions. Assume an average size of 1024 bytes of storage for each ICL record. With an allocation of `CYL(3,1)`, the data set can have a maximum of 125 cylinders on a single 3390 volume for a total size of 105 MB. This would mean that the data set should be able to hold at least 102,500 certificates. If multiple volume support is used, you can double this amount.

If your anticipated needs differ greatly from the preceding value, you need to adjust the space allocation parameters `CYL(3,1)` on the DEFINE CLUSTER statement for the ICL. (This is the second DEFINE CLUSTER statement in IKYCVSV1. See "IKYCVSV1" on page 665 for a code sample of this file.) You might also want to proportionally adjust the space allocation parameters `TRK(5,1)` on the DEFINE ALTERNATEINDEX statements for the ICL. These are defined in the DEFALTDX job step. (Their names contain the `icl` qualifier.)

Determining storage needs for the object store

Object store records are not permanent. They are deleted when they are no longer needed. Unlike the ICL, the object store does not grow beyond a certain point, unless there is a sharp increase in certificate request activity. Typically, a certificate request record is less than twice the size of the ICL record. Assume that one object store record and its companion posting record occupy a total of 2560 bytes of storage. With a space allocation of `CYL(3,1)`, the data set can have a maximum of 125 cylinders on a single 3390 volume for a total size of 105 MB, which would hold at least 41,000 concurrent certificate requests. If multiple volume support is used, you can double this amount.

If your anticipated needs differ greatly from the preceding value, you need to adjust the space allocation parameters `CYL(3,1)` on the DEFINE CLUSTER statement for the object store. (This is the first DEFINE CLUSTER statement in IKYCVSV1. See "IKYCVSV1" on page 665 for a code sample of this file.) You might also want to proportionally adjust the space allocation parameters `TRK(5,1)` on the DEFINE ALTERNATEINDEX statements for the object store. These are defined in the DEFALTDX job step. (Their names contain the `ost` qualifier.)

If you set the `RemoveInactiveReqs` and `RemoveCompletedReqs` parameters in the configuration file to `0`d, certificate request records are not deleted and you should adjust the space allocation accordingly.

(Optional) preliminary steps for establishing VSAM RLS

Your team can configure PKI Services to take advantage of a Parallel Sysplex environment. This enables you to start multiple instances of the PKI Services daemon (one per image) that work in unison. The daemons are totally independent of each other, but they all act upon a single common data store containing the ICL and object store VSAM data sets.

If you want to run multiple instances of PKI Services in a Parallel Sysplex (one per image), you must first establish the data sharing environment suitable for VSAM record-level sharing (RLS).

Before you begin

The following steps assume that the coupling facility is already set up. If not, see *z/OS MVS Programming: Sysplex Services Guide* for information about how to set up the coupling facility. Also, see "(Optional) steps for enabling existing PKI Services VSAM data sets for VSAM RLS" on page 119 for additional information about setting up VSAM data sets to run PKI Services in a sysplex.

Procedure

Perform the following steps to establish VSAM RLS. For specific information on how to perform these steps, see the topic about administering VSAM record level sharing in *z/OS DFSMSdfp Storage Administration*.

1. Define and activate at least two sharing control data sets (SHCDS) and one spare SHCDS for recovery purposes.

   ____
2. Define CF lock structure to MVS.

   ____
3. Define CF lock structure in the SMS base configuration.

   ____
4. Define at least one storage class for VSAM record-level sharing (RLS).

   You must record the name of this storage class for use in creating the VSAM data sets for PKI Services.

   Table 26. VSAM RLS information you need to record

VSAM information you need to record	Your value
Name of storage class for VSAM RLS	

   ____

Steps for creating the VSAM object store and ICL data sets and indexes

You need to perform this task only if you are configuring PKI Services for the first time.

PKI Services uses VSAM data sets to store requests in progress and issued certificates. You need to create these data sets manually.

If you expect the object store or ICL data sets to exceed 4 GB in size, consider using VSAM extended addressability. For more information, see *z/OS DFSMS Using Data Sets*.

Before you begin

If you also want to run multiple instances of PKI Services in a Parallel Sysplex (one per image), you need to have performed the steps that are described in "(Optional) preliminary steps for establishing VSAM RLS" on page 117.

Procedure

Perform the following steps to create the VSAM object store and ICL data sets and indexes (if you are configuring PKI Services for the first time):

1. Copy the sample JCL in `SYS1.SAMPLIB(IKYCVSV1)` to your JCL data set. (See "IKYCVSV1" on page 665 for a code sample of this file.)

   ____
2. Update your data set as directed in the instructions in the prolog of the sample JCL:

   a. Change the JOB card.

   b. Change the VOL statements.

- If you are running multiple instances of PKI Services in a Parallel Sysplex, replace the VOL statements with STORCLAS statements that specify the storage class recorded in Table 26 on page 118, for example:

  ```
  STORCLAS(VSAMRLS)
  ```

- If you are running without a Parallel Sysplex, replace the `vvvvvv` in the VOL statements with a VOL=SER suitable for your VSAM data sets.

c. If you are running multiple instances of PKI Services in a Parallel Sysplex, remove the SPANNED and CISIZE statements in the file. Make sure that you do not delete a closing parenthesis that is still needed. For example, change:

```
DATA -
(NAME(PKISRVD.VSAM.ICL.DA) -
CISZ(4096) -
SPANNED) -
```

to

```
DATA -
(NAME(PKISRVD.VSAM.ICL.DA)) -
```

d. You can optionally change the data set names but must remember to make equivalent changes in the `pkiserv.conf` file if you do so. (See Step "1.d" on page 89.)

e. Update the primary and secondary extent allocations based on your anticipated future needs. (See "Planning VSAM storage requirements" on page 116 for guidelines on determining the space that you need.) These are the default allocations for each type of data set:

Base cluster
`CYL(3,1)`

Alternate indexes
`TRK(5,1)`

Guideline: Do not change any numeric values, other than the primary and secondary space allocation values for the base cluster and alternate index data sets.

____

3. Submit the job when your changes are complete.

____

(Optional) steps for enabling existing PKI Services VSAM data sets for VSAM RLS

To run PKI Services in parallel, the UNIX programmer must specify `SharedPLEX=T` in the `pkiserv.conf` configuration file. (See the `SharedPLEX=T` row in Table 21 on page 74.) The MVS programmer enables the sysplex to access the VSAM data sets.

Before you begin

You need to have performed the steps that are described in "(Optional) preliminary steps for establishing VSAM RLS" on page 117.

Procedure

Perform the following steps to enable your existing PKI Services data sets for VSAM record-level sharing (RLS).

1. Copy the sample reallocation JCL in `SYS1.SAMPLIB(IKYRVSAM)` or `SYS1.SAMPLIB(IKYRVSV1)` to your JCL data set.

   Attention: Do not confuse IKYCVSV1 with IKYRVSV1, because the former destroys your existing VSAM data sets.

____

2. Update your data set, following the instructions in the prolog of the sample JCL:
   a. Change the JOB card.
   b. Change the STORCLAS statements.

      Note: Before submitting this job, check your access control service (ACS) routines for naming convention definitions that are used to create the VSAM RLS data sets.
   c. Rename the source data sets to the names of your existing object store and ICL data sets.
   d. Change the destination data set names.

      Note: Remember to give the UNIX programmer the data set names so the UNIX programmer can make equivalent changes in the `pkiserv.conf` file. See "(Optional) Steps for updating the configuration file" on page 74.
   e. Update the primary and secondary extent allocations based on your anticipated future needs. (See "Planning VSAM storage requirements" on page 116 for guidelines on determining the space that you need.) These are the default allocations for each type of data set:

      Base cluster
         `CYL(3,1)`

      Alternate indexes
         `TRK(5,1)`

      Guideline: Do not change any numeric values, other than the primary and secondary space allocation values for the base cluster and alternate index data sets.

   ____
3. Submit the job when your changes are complete.

   ____

Tuning VSAM performance

Depending on your environment, your VSAM performance might be improved by providing buffer space for the VSAM data sets as part of the IKYSPROC (alias PKISERVD) started procedure.

(Optional) steps for adding VSAM buffer space

Perform the following steps to add buffer space for the VSAM data sets as part of the IKYSPROC (alias PKISERVD) started procedure. When completed, you need to stop and restart PKI Services before your changes take effect.

1. Make a backup copy of `SYS1.PROCLIB(PKISERVD)`. (See "PKISERVD sample procedure to start PKI Services daemon" on page 682 for a code sample of this file.)

   ____
2. Edit the JCL in `SYS1.PROCLIB(PKISERVD)` in the following ways.
   a. Append the DD statements that are shown in the following example to the end of the PKISERVD procedure.
   b. Change the names of the VSAM data sets shown in the example to the names that you used when you executed the IKYCVSV1 sample job to allocate the data sets. (See "Steps for creating the VSAM object store and ICL data sets and indexes" on page 118.)

   Example:

```
//* ObjectDSN data set
//OST      DD  DSN=PKISRVD.VSAM.OST,DISP=SHR,
//  AMP=('BUFNI=6,BUFND=4')
//* ObjectTidDSN data
//TID      DD  DSN=PKISRVD.VSAM.OST.PATH,DISP=SHR,
//  AMP=('BUFNI=6,BUFND=4')
//* ObjectStatusDSN data set
//OSTAT    DD  DSN=PKISRVD.VSAM.OST.STATUS,DISP=SHR,
//  AMP=('BUFNI=1,BUFND=4')
//* ObjectRequestorDSN data set
```

```
//OREQ     DD  DSN=PKISRVD.VSAM.OST.REQUESTR,DISP=SHR,
//  AMP=('BUFNI=1,BUFND=4')
//* ICLDSN data set
//ICL      DD  DSN=PKISRVD.VSAM.ICL,DISP=SHR,
//  AMP=('BUFNI=6,BUFND=4')
//* ICLStatusDSN data set
//ISTAT    DD  DSN=PKISRVD.VSAM.ICL.STATUS,DISP=SHR,
//  AMP=('BUFNI=1,BUFND=4')
//* ICLRequestorDSN data set
//IREQ     DD  DSN=PKISRVD.VSAM.ICL.REQUESTR,DISP=SHR,
//  AMP=('BUFNI=1,BUFND=4')
```

____

3. Edit the names in the new DD statements to match the data sets in the **ObjectStore** section of the PKI Services configuration file (`pkiserv.conf`).

____

4. Optionally, you might need to adjust the numeric values for BUFNI and BUFND.

   Tip: The STATUS and REQUESTR data sets are accessed sequentially only, while the others are accessed both sequentially and directly. Keep this in mind when adjusting the values. (For more information about VSAM buffer space, see *z/OS DFSMS Using Data Sets*.)

____

5. Edit the **ObjectStore** section of the PKI Services configuration file (`pkiserv.conf`) to change the existing data set names to the following DD names:

Default value	Suggested value
`ObjectDSN='pkisrvd.vsam.ost'`	`ObjectDSN=DD:OST`
`ObjectTidDSN='pkisrvd.vsam.ost.path'`	`ObjectTidDSN=DD:TID`
`ObjectStatusDSN='pkisrvd.vsam.ost.status'`	`ObjectStatusDSN=DD:OSTAT`
`ObjectRequestorDSN='pkisrvd.vsam.ost.requestr'`	`ObjectRequestorDSN=DD:OREQ`
`ICLDSN='pkisrvd.vsam.icl'`	`ICLDSN=DD:ICL`
`ICLStatusDSN='pkisrvd.vsam.icl.status'`	`ICLStatusDSN=DD:ISTAT`
`ICLRequestorDSN='pkisrvd.vsam.icl.requestr'`	`ICLRequestorDSN=DD:IREQ`

____

6. Save your changes.

____

7. Stop and restart PKI Services.

____

Backing up and restoring the VSAM data sets

You can use the DFSMSdss DUMP and RESTORE commands to back up and restore the VSAM data sets. For information about these commands, see *z/OS DFSMSdss Storage Administration*.

Steps for backing up the VSAM data sets

Perform the following steps to back up the VSAM data sets using the DFSMSdss DUMP command.

Procedure

1. Copy the sample JCL in `SYS1.SAMPLIB(IKYVBKUP)` to your JCL data set. (See "IKYVBKUP" on page 680 for a code sample of this file.)
2. Update the JCL as directed in the instructions in the prolog of the sample JCL:
   a) Change the job card.

   b) If you are not using the default data set qualifiers, change all occurrences of "PKISRVD.VSAM" to the qualifiers you are using.
   c) Change all occurrences of *vvvvvv* to a VOLSER value that contains sufficient free space to contain a complete backup of both of the PKI Services VSAM data set clusters.
   d) Change the primary and secondary allocation values for the backup data set to values that ensure a complete backup of both VSAM data set clusters. Change the *xxx* value for the primary allocation, and the *yyy* value for the secondary allocation.
   e) You can optionally change the data set name for the BACKUPDS DD, but if you do be sure to make the same change to the restore JCL in IKYVREST.
3. Stop the PKI Services address space. For information about how to do this, see Chapter 10, “Starting and stopping PKI Services,” on page 129.
4. Submit the job.
5. When the job has completed, restart the PKI Services address space. For information about how to do this, see Chapter 10, “Starting and stopping PKI Services,” on page 129.

Results

When you are done, you have created a backup data set containing the PKI Services VSAM files.

Steps for restoring the VSAM data sets

Perform the following steps to restore the VSAM data sets using the DFSMSdss RESTORE command.

Procedure

1. Copy the sample JCL in `SYS1.SAMPLIB(IKYVREST)` to your JCL data set. (See “IKYVREST” on page 681 for a code sample of this file.)
2. Update the JCL as directed in the instructions in the prolog of the sample JCL:
   a) Change the job card.
   b) If you are not using the default data set qualifiers, change all occurrences of "PKISRVD.VSAM" to the qualifiers you are using.
   c) If you changed the default data set name for the BACKUPDS DD in the IKYVBKUP JCL, change the data set name (DSN) value in the BACKUPDS DD statement to match.
3. Stop the PKI Services address space. For information about how to do this, see Chapter 10, “Starting and stopping PKI Services,” on page 129.
4. Submit the job.
5. When the job has completed, restart the PKI Services address space. For information about how to do this, see Chapter 10, “Starting and stopping PKI Services,” on page 129.

Results

When you are done, you have restored the PKI Services VSAM files from a backup data set.

Creating the object store and ICL using Db2 tables

PKI Services can use Db2 tables to store requests in progress and issued certificates. You need to create these tables (see “Steps for creating the object store and ICL Db2 tables” on page 123).

Notice that in Version 2 Release 3, the following two versions of DBRMs are shipped.

- `Version 0` called IKYPDBRM
- `Version 1` called IKYPDBR1

The versions are indicated in the `MEMBER` parameter in the sample IKYSBIND job. IKYPDBR1 is required for `version 1` object store and ICL. If you need to build a `version 0` package, use IKYPDBRM.

Sysplex considerations

Requirements: These requirements apply when you share the Db2 tables in a sysplex environment:

- If PKI Services is set up to run in a sysplex environment and share the Db2 tables for the object store and ICL, all systems in the sysplex must be running the same release of z/OS, which must be z/OS Version 1 Release 13 or later.
- Db2 Version 9 must be installed and the Db2 subsystems that share data must belong to a Db2 data sharing group that runs on a sysplex cluster.
- The `SharedPLEX` parameter in the configuration file `pkiserv.conf` must be set to T (true). (See the `SharedPLEX`=T row in Table 21 on page 74.)

If you want to enable the SCEP requester enhancement, ensure that all members are migrated to Version 2 Release 3. The migration to Version 2 Release 3 must be complete before the conversion of OST and ICL to the new version for each instance of PKI. If the PKI instance starts successfully from all of the members, you can start to convert the OST and ICL for each instance with the following steps:

- Run IKYCDBV1 to create the new Db2 Object Store and ICL tables.
- Run IKYSBIND by using the `version 1` DBRM and IKYPDBR1 to build the new package with a new name.
- Stop the instance of PKI from all members.
- Run the conversion utility, `db2conv`.
- Update `pkiserv.conf` to specify the `DBVersion` to 1 from all members.
- Start the instance of PKI from all members.

Planning Db2 storage requirements

The space allocation for Db2 is specified in the CREATE TABLESPACE SQL statements in the IKYCDBV1 sample. The PRIQTY value is `144400` in kilobytes, which holds 144,400 ICL records that are 1024 bytes in length. Assuming that an object store certificate request record is approximately twice the size of an ICL record, 144,400 kilobytes holds 72,200 object store records. No SECQTY keyword is specified, because Db2 calculates a secondary quantity value.

If your anticipated needs differ greatly from the amount of storage provided by the PRIQTY keyword in the IKYCDBV1 sample, adjust the PRIQTY value.

If you set the `RemoveInactiveReqs` and `RemoveCompletedReqs` parameters in the configuration file to `0`d, certificate request records are not deleted and you should adjust the space allocation accordingly.

Steps for creating the object store and ICL Db2 tables

Perform the following steps to implement the object store and ICL using Db2 tables.

Before you begin

- Db2 must be installed and running.
- You need to know the name of the Db2 subsystem or the group attachment name for the object store and ICL.
- You need to know the name you are using for the Db2 package. The default package name is `MASTERCA`.
- You need to know the name of the PKI Services daemon.

About this task

You need to perform this task only if you are configuring PKI Services for the first time. If you have been using VSAM files for the object store and ICL and want to use Db2 tables instead, follow the steps in "Converting the object store and ICL from VSAM to Db2" on page 125.

Procedure

1. Update the IKYCDBV1 sample and run it to create the Db2 objects. This sample contains SQL statements that define the database, table space, tables, and indexes for the ICL and object store.
   a) Copy the sample from `SYS1.SAMPLIB(IKYCDBV1)`. (For a code sample of this file, see "IKYCDBV1" on page 658.)
   b) As directed in the sample, if you have run the sample before, uncomment the statements in the first section so that the existing indexes, tables, table spaces, and database are dropped and committed.
   c) If you are using a package name other than the default name `MASTERCA`, change every occurrence of `MASTERCA` in the sample to the package name you have chosen.
   d) Run your updated copy of IKYCDBV1. You can use the Db2 SPUFI facility to run it.

   ____

2. Update the sample job IKYSBIND and run it to build the PKI Services package using the appropriate DBRM version.
   a) Copy the sample job from `SYS1.SAMPLIB(IKYSBIND)`. (For a code sample of this file, see "IKYSBIND" on page 677.)
   b) If you are using a package name other than the default name `MASTERCA`, change every occurrence of `MASTERCA` in the sample job to the package name you have chosen.
   c) Run your updated copy of IKYSBIND.

   ____

3. Update the sample job IKYSGRNT and run it to grant execute privilege on the Db2 package for PKI Services to the PKI Services daemon user ID.
   a) If the PKI Services daemon has a name other than the default name of `PKISRVD`, change `PKISRVD` to the name of your PKI Services daemon. (The daemon name is determined by the `daemon` variable in IKYSETUP. For more information, see Chapter 4, "Running IKYSETUP to perform RACF administration," on page 39.)
   b) Change `MASTERCA` to the package name that was used in the IKYSBIND job.
   c) Run your updated copy of IKYSGRNT.

   ____

4. Update the IKYSETUP REXX exec to indicate that you are using Db2 tables, so that IKYSETUP gives the PKI Services daemon the access it needs to the Db2 Resource Recovery Services Access Facility (RRSAF). In "IKYSETUP sample" on page 624, see Question 6 for information about using Db2 as the repository for the Issued Certificate List (ICL) and Object Store. Change the line:

   ```
   db2_repos = 0
   ```

   to:

   ```
   db2_repos = 1
   ```

   Update the line:

   ```
   db2_subsys = 'DSN9'
   ```

   and change `'DSN9'` to the name of the Db2 subsystem or the group attachment name that provides the repository. Run the updated copy of IKYSETUP.

   ____

5. Update the configuration file, `pkiserv.conf`, to indicate that you are using Db2 tables.
   - In the **ObjectStore** section, uncomment the following line:

     ```
     # DBType=DB2
     ```

and make sure that the following line is commented out or omitted:

```
# DBType=VSAM
```

- In the **ObjectStore** section, uncomment the following line:

  ```
  # DBPackage=MasterCA
  ```

  If you used a package name other than `MasterCA` in the IKYCDBV1 and IKYSBIND samples, change `MasterCA` in this line to the package name you used.
- In the **ObjectStore** section, uncomment the following line:

  ```
  # DBSubsystem=DSN9
  ```

  If you are using a Db2 subsystem or a group attachment with a name other than DSN9, change DSN9 in this line to the name of your Db2 subsystem or group attachment.
- If the object store and ICL are shared in a sysplex with other instances of PKI Services, change the line:

  ```
  SharedPLEX=F
  ```

  to

  ```
  SharedPLEX=T
  ```

- Restart PKI Services so that your changes take effect.

  ____

Results

When you are done, you have created Db2 objects for the object store and ICL.

Converting the object store and ICL from VSAM to Db2

If you have previously set up PKI Services to use VSAM data sets for the object store and ICL, you can convert it to use Db2 tables instead. To do this you must copy the data in your existing VSAM data sets to Db2 tables.

Steps for converting the object store and ICL from VSAM to Db2

Perform the following steps to convert the object store and ICL from VSAM data sets to Db2 tables.

Before you begin

- Db2 must be installed and running.
- You need to know the name of the Db2 subsystem or the group attachment name.
- You need to provide the name of the Db2 package that accesses the Db2 database using the version that corresponds to the version of the VSAM storage that you are converting. The Db2 package name is assigned when IKYSBIND is run. If you are converting a `version 0` VSAM object store and ICL, use the Db2 package name that is assigned when IKYSBIND is used to create a package for the IKYPDBRM Db2 DBRM. If you are converting `version 1` VSAM storage, use the Db2 package name that is assigned when IKYSBIND is used to create a package for the IKYPDBR1 Db2 DBRM.
- You need to know the name of the PKI Services daemon.

Procedure

1. Stop the instance of PKI Services for which you want to do the conversion. For information about how to do this, see Chapter 10, “Starting and stopping PKI Services,” on page 129.

   ____

2. Update the IKYCDBV1 sample, for `version 1`, or the IKYCDB2 sample, for `version 0`. Run it to create the Db2 objects. See step "1" on page 124.

   ____
3. Update the sample job IKYSBIND to point to IKYPDBRM or IKYPDBR1 DBRM and run it to create the `version 0` or `version 1` Db2 package. See step "2" on page 124.

   ____
4. Update the sample job IKYSGRNT and run it to grant execute privilege on the Db2 package for PKI Services to the PKI Services daemon user ID. See step "3" on page 124.

   ____
5. Use the utility program `vsam2db2` to copy the data from your VSAM data sets to the corresponding Db2 tables. See "Using the vsam2db2 utility" on page 448.

   ____
6. Update the configuration file, `pkiserv.conf`, to indicate that you are using Db2 tables. See step "5" on page 124.

   ____
7. Restart PKI Services. For information about how to do this, see Chapter 10, "Starting and stopping PKI Services," on page 129.

   ____

Results

When you are done, you have copied your object store and ICL VSAM data sets to Db2 tables, and set up PKI Services to use the Db2 tables.

NOT Programming Interface Information

Columns in the ICL and object store Db2 tables

The information in Table 27 on page 126 and Table 28 on page 127 is intended for your use when creating Db2 queries and reports. It is not a programming interface.

Table 27. Columns in the object store Db2 tables

Column name	**Format**	**Contents**
Header:		
RECORD_KEY	BINARY(4)	
RECORD_STATE	BINARY(4)	
REQDATA_LEN	INTEGER	
REQUESTOR_NAME	VARCHAR(32)	Requestor name
TRANS_ID	CHAR(24)	Transaction ID
COMMENT	VARCHAR(64)	Comment
ISSUED_TIME	TIMESTAMP	Issued time
LAST_CHANGE_TIME	TIMESTAMP	Last changed time
TEMPLATE_NICKNAME	VARCHAR(8)	Template nickname
SERIAL_NUM	BINARY(4)	Serial number
DB_VERSION	BINARY(1)	Version record of this object store
SCEP_TRANSID	VARCHAR(128)	SCEP Transaction ID

Table 27. Columns in the object store Db2 tables (continued)

Column name	Format	Contents
Body:		
REQDATA	VARBINARY(32380)	Request object in ASN1 format

Table 28. Columns in the ICL Db2 tables

Column name	Format	Contents
Header:		
SERIAL_NUM	BINARY(4)	Serial number
CERT_STATE	BINARY(4)	
CERT_LEN	INTEGER	
REQUESTOR_NAME	VARCHAR(32)	Requestor name
REVOKE_DATE	TIMESTAMP	Revocation date
INVALID_DATE	TIMESTAMP	Expiration date
REVOKE_REASON	INTEGER	Revoke reason
COMMENT	VARCHAR(64)	Comment
ISSUED_TIME	TIMESTAMP	Issued time
LAST_CHANGE_TIME	TIMESTAMP	Last changed time
TEMPLATE_NICKNAME	VARCHAR(8)	Template nickname
OBFUS_PW	VARBINARY(33)	Obfuscated passphrase
PROCESS_FLAGS	BINARY(4)	Flags
KEYID	BINARY (20)	SHA1 hash of public key
CRLDP_NUM	INTEGER	CRL DP number
EXPIRE_EPOCH_DAYS	INTEGER	Expiration date, expressed as "days since the epoch " (January 1, 1970)
EXPIRE_DATE	TIMESTAMP	Expiration date, expressed in Db2 TIMESTAMP format
KU_DIGTSIG	BINARY(1)	digitalSignature keyusage
KU_NONRPU	BINARY(1)	nonRepudiation keyusage
KU_KEYENC	BINARY(1)	keyEncipherment keyusage
KU_DATAENC	BINARY(1)	dataEncipherment keyusage
KU_KEYAGR	BINARY(1)	keyAgreement keyusage
KU_CRTSGN	BINARY(1)	keyCertSign keyusage
KU_CRLSGN	BINARY(1)	CRLSign keyusage
KU_ENCONLY	BINARY(1)	encipherOnly keyusage
KU_DECONLY	BINARY(1)	decipherOnly keyusage
EKU_SEVAUTH	BINARY(1)	serverAuth extended keyusage

Table 28. Columns in the ICL Db2 tables (continued)

Column name	Format	Contents
EKU_CLIAUTH	BINARY(1)	clientAuth extended keyusage
EKU_CODESGN	BINARY(1)	codeSigning extended keyusage
EKU_EMLPROT	BINARY(1)	emailProtection extended keyusage
EKU_TMESTMP	BINARY(1)	timeStamping extended keyusage
EKU_OCSPSGN	BINARY(1)	OCSPSigning extended keyusage
EKU_MSSCLNON	BINARY(1)	Microsoft smart card logon extended keyusage
PREV_SERIAL_NUM	BINARY(4)	Previous serial number if the certificate has been renewed
SUBJ_DN	VARCHAR(1024)	Subject distinguished name
DB_VERSION	BINARY(1)	Version record of this ICL
SCEP_TRANSID	VARCHAR(128)	SCEP Transaction ID
Body:		
X509CERT	VARBINARY(10240)	Certificate in ASN1 format

End NOT Programming Interface Information

Chapter 10. Starting and stopping PKI Services

You start the PKI Services daemon or daemons the first time you are configuring PKI Services or if you are adding sysplex support to run multiple independent instances of PKI Services (one per image) on a sysplex. The MVS programmer performs these tasks.

Steps for starting the PKI Services daemon

You need to start the PKI Services daemon if:

- You are configuring PKI Services for the first time.
- You want to enable Simple Certificate Enrollment Protocol (SCEP) or Enrollment over Secure Transport protocol (EST).
- You renewed your CA or RA certificate.
- You want to use Parallel Sysplex support and need to run another instance of the PKI Services on a different image in the sysplex.
- You stopped PKI Services and need to restart it.
- You created a new (additional) CA domain and want to start it.
- The PKI Services CA certificate private key is managed by ICSF and ICSF became unavailable. After you fix the problem with ICSF, you need to stop and restart PKI Services.
- You use Db2 tables for the object store and ICL, and you need to stop Db2. You must stop PKI Services first. After Db2 restarts, restart PKI Services.

Before you begin

- ICSF must be operational.
- If you use Db2 tables for the object store and ICL, you need to start Db2.
- Your HTTP server should be SSL-enabled (see Chapter 7, “Updating IBM HTTP Server - Powered by Apache configuration and starting the server,” on page 101) and the uncustomized PKISERV application should be ready for use.
- If you are starting PKI Services for the first time, you need to know the runtime directory, called `runtime-dir` in the command examples. The default is `/etc/pkiserv/`. The MVS programmer was asked to record any changes to the default; see Table 3 on page 11.
- If you are starting PKI Services for a new CA domain, you need to know the job name that contains the instance of the daemon you need to start. Do not use these steps. Instead, see “Adding a new CA domain” on page 302 for steps to add a new domain and start the new daemon.

Procedure

Perform the following steps to start the PKI Services daemon and view your web pages:

1. If you have not done so already, start ICSF, the web server and the LDAP server. You must also start Db2 if your implementation uses it.

   ____

2. If you want to test the configuration to this point before customizing PKI Services (preferred), you need to temporarily prevent PKI Services from posting issued certificates to LDAP because posting to LDAP is not successful. Have the UNIX programmer perform the following steps to prevent PKI Services from posting issued certificates to LDAP:

   a. Edit the PKI Services configuration file (by default, this is: `/etc/pkiserv/pkiserv.conf`).

   ____

b. Set `NumServers=0` in the **LDAP** section of the file.

____

c. Exit to save your changes.

Note: After testing the configuration, you need to stop PKI Services and undo the change in this step (see Step “5” on page 130) and then restart PKI Services.

____

3. Start the PKI Services daemon from the MVS console by entering the following command:

```
S PKISERVD
```

Notes:

a. You must start the PKI Services daemon only from a started procedure. PKI Services rejects all other methods of starting the daemon (including INETD, `/etc/rc`, UNIX shell, or submitted JCL job).

b. Depending on the amount of customization you did, there are various versions of the preceding command to start the PKI Services daemon. For example, if you changed the `pkiserv.envars` file (see Step “7” on page 70), you need to specify its new location as a parameter in the START command:

```
S PKISERVD,DIR='runtime-dir'
```

(Single quotation marks are required to maintain the character case of the values being assigned to the substitution parameters.)

The command in the following example specifies the runtime directory and the file name of the environment variables file:

Example:

```
S PKISERVD,DIR='/etc/pkiserv',FN='pkiserv.envars'
```

The default time zone is EST5EDT. If you need to change this, you can supply the new value as a parameter, as in the following examples:

Examples:

```
S PKISERVD,TZ=PST8PDT
S PKISERVD,JOBNAME=jobname,DIR='/etc/pkiserv',FN='pkiserv.envars',TZ=PST8PDT
```

____

4. Go to your web pages by entering the following URL from your browser:

```
http://webserver-fully-qualified-domain-name/PKIServ/ssl-cgi/camain.rexx
```

The *`webserver-fully-qualified-domain-name`* is the common name (CN) portion of the web server's distinguished name; see Table 11 on page 41.

You should be able to go through your web pages to request, retrieve, and revoke a certificate of type "PKI browser certificate for authenticating to z/OS". Ensure you can do this before trying to customize the application.

____

5. If you elected to test the configuration, you need to stop PKI Services (see “Stopping the PKI Services daemon” on page 131), undo the change in Step “2” on page 129, and then restart PKI Services. To undo the change in Step “2” on page 129:

a. Edit the PKI Services configuration file (by default `/etc/pkiserv/pkiserv.conf`).

b. Set `NumServers=`*n* in the **LDAP** section of the file, where *n* is the same number of LDAP servers indicated in Table 25 on page 108.

c. Exit to save your changes.

____

Stopping the PKI Services daemon

To stop the PKI Services daemon, you can use either the MODIFY (or **F**) console command, or the STOP (or **P**) command. Enter one of the following two commands:

```
F PKISERVD,STOP
```

or

```
P PKISERVD
```

Part 3. Customizing PKI Services

PKI Services provides two ways to implement and customize the PKI Services web interface:

- Using REXX CGI scripts. This method uses a text template file.
- Using JavaServer pages (JSPs). This method uses an XML template file.

Both methods produce identical web pages with the default files shipped with PKI Services.

This part includes the following topics:

- Chapter 11, "Customizing the end-user web application if you use REXX CGI execs," on page 135 provides an overview of the `pkiserv.tmpl` file, which contains the certificate templates, and explains customizing the end-user web pages when you are implementing them using the REXX CGI execs.
- Chapter 12, "Customizing the administration web pages if you use REXX CGI execs," on page 233 provides an overview of the REXX CGI execs and explains how to customize the administration web pages when you are implementing them using the REXX CGI execs.
- Chapter 13, "Implementing the web application using JavaServer pages," on page 237 provides an overview of customizing the PKI Services web pages when you implement them using the JavaServer pages.
- Chapter 14, "Advanced customization," on page 283 explains:
  - "Scaling for high volume installations" on page 283
  - "Using certificate policies" on page 284
  - "Updating the signature algorithm" on page 287
  - "Customizing distribution point CRLs" on page 290
  - "Creating a distribution point ARL" on page 295
  - "Enabling support for large CRLs" on page 297
  - "Using the OCSP responder" on page 298
  - "Adding an application domain" on page 298
  - "Adding a new CA domain" on page 302
  - "Customizing email notifications sent to users" on page 317
  - "Setting up automatic renewal of certificates" on page 323
  - "Setting up PKI Services to generate keys for certificate requests" on page 325
  - "Adding custom extensions to certificates" on page 328
- Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333
- Chapter 16, "Using Enrollment over Secure Transport (EST)," on page 341
- Chapter 17, "Customizing with installation exit routines," on page 349 explains how to use installation exits with PKI Services.

Chapter 11. Customizing the end-user web application if you use REXX CGI execs

This information applies if you are using REXX CGI execs for your PKI Services web pages. If you are using JavaServer pages (JSPs), see Chapter 13, “Implementing the web application using JavaServer pages,” on page 237.

For certificate processing to work, you need to customize the end-user web pages at least to some degree. Before you begin to customize web pages, you need to understand the `pkiserv.tmpl` certificate templates file. This file contains certificate templates, which define the fields that comprise a specific certificate request. This topic describes the `pkiserv.tmpl` certificate templates file and explains how to use it to customize the end-user web pages. This topic also explains the relationship between CGIs and the certificate templates file. Finally, this topic also contains information about customizing email notifications. (Sending email notifications is an optional feature.)

Contents of the pkiserv.tmpl certificates templates file

The `pkiserv.tmpl` certificate templates file contains certificate templates that define the fields that comprise a specific certificate request. The file contains a mixture of true HTML and HTML-like tags. The HTML can contain JavaScript for input field verification.

The main sections of the `pkiserv.tmpl` certificate templates file are listed in Table 29 on page 135:

Table 29. Structure and main divisions of the certificate template file (pkiserv.tmpl)

Structure and main divisions of the certificate template file (pkiserv.tmpl)
A prolog section of comments explaining main sections, subsections, named fields, and substitution variables.
A DEBUG flag appears directly after the prolog section. (You can change `DEBUG=0` to `DEBUG=1` to get CGI debugging information.)
APPLICATION sections The APPLICATION sections contain subsections, which produce certain web pages, such as the PKI Services home page shown in “Steps for accessing the end-user web pages” on page 376. For details, see “The APPLICATION sections” on page 147.
TEMPLATE sections These are the certificate templates (models) that contain the HTML to produce certificate request forms. They also define the fields that are permissible in the certificate. For details, see “TEMPLATE sections” on page 151.
INSERT sections These contain HTML for certain web pages, such as the "Request submitted successfully" web page, and certificate field dialogs, such as text entry boxes (the common name INSERT produces a text box where the user enters this information) and drop-downs. For details, see Figure 46 on page 389.

The `pkiserv.tmpl` file begins with a prolog. This is a section of comments that explains the main sections and subsections of the file. Any line with a # in column 1 is a comment.

Only the APPLICATION sections and TEMPLATE sections can contain subsections, but all three can contain named fields and substitution variables.

What are substitution variables?

A substitution variable holds a value that HTML code can reference. At run time, the actual value replaces a substitution variable.

You use square brackets to delineate a substitution variable.

Example:

```
[base64cert]
```

Notes:

1. Substitution variables are case-sensitive.
2. Depending on the section where a substitution variable is present, it might not have a valid meaning. For example, the `[base64cert]` substitution variable is meaningless before the certificate is retrieved. Therefore, in this case, the value of `[base64cert]` would be the null string (an empty string).

Table 30 on page 136 summarizes valid substitution variables:

Table 30. Substitution variables

Substitution variable	Description
`altrawvalue`	The concatenated value of the AltOther fields.
`base64cert`	The requested certificate, base64-encoded.
`browsertype`	A special substitution variable to qualify named fields only. It enables Mozilla-based browsers and Internet Explorer to perform browser-specific operations, such as generating a public and private key pair. To generate a key pair, a Mozilla-based browser uses a KEYGEN HTML tag, while Internet Explorer uses ActiveX controls. An example is if %%PublicKey[browsertype]%% is specified in a TEMPLATE CONTENT section. If the user referencing this section uses the Mozilla Firefox browser, INSERT PublicKeyNS is included. If the browser that is used is Microsoft Internet Explorer, INSERT PublicKeyIE is included.
`cadomain`	The CA domain name that is used to examine the preregistration record in the Simple Certificate Enrollment Protocol (SCEP).
`errorinfo`	Information such as the return code and reason code related to a failing SAF call.
`iecert`	The requested certificate in a form that Microsoft Internet Explorer accepts.
`keyid`	The SHA1 hash of the generated public key.
`optfield`	A special substitution variable that should be placed in any certificate field name INSERT where the end user can supply the value. It makes the input field optional.
`p12cert`	A PKCS #12 package in DER encoded format, containing a certificate and the private key that PKI Services generated for the certificate.
`printablecert`	This variable contains the certificate details so that the end user can confirm that the certificate is the correct one to renew or revoke. The displayed data is extracted from the ICL entry.

Table 30. Substitution variables (continued)

Substitution variable	Description
readonly	This substitution variable is converted to a null string when the INSERT is used for input purposes, such as when requesting a certificate. It is substituted with the string `readonly` when the INSERT is used for output purposes, such as when displaying request or certificate information.
requestor	The email address of the requestor, when the keys for the certificate were generated by PKI Services.
serialno	The serial number of a certificate.
tmplname	A certificate template name. This is primed from the HTML tag <SELECT NAME="Template"> in the <APPLICATION NAME=PKISERV> section. The end user selects it on the first web page.
transactionid	A unique value that is returned from a certificate request.

What are named fields?

Named fields insert common HTML code, such as a common input field or a page header or footer, in a web page. (Each named field refers to a corresponding INSERT section.) A named field is delineated with %%.

Examples:

```
%%Country%%
%%-pagefooter%%
```

Note: Named fields are case-sensitive.

A named field can include or not include a dash. A named field without a dash, such as `%%Label%%` might have a special meaning as a certificate field. Its special meaning depends on the section in which it appears. (See "Relationship between CGIs and the pkiserv.tmpl file" on page 217 for more information.)

A named field with a dash, such as `%%-pagefooter%%`, has no special meaning. PKISERV treats it simply as HTML code to insert. Any special meaning the named field might have, based on the section in which it is contained, is ignored. For example, in a TEMPLATE CONTENT section (see "TEMPLATE sections" on page 151) if you specify `%%-pagefooter%%`, `-pagefooter` is not considered a certificate field name. However, the INSERT section with the name `-pagefooter` is included in the HTML page displayed to the end user.

INSERT sections

Although the INSERT sections are at the end of the `pkiserv.tmpl` certificate templates file, they are explained first because of their relationship to named fields. Any named field that is used in the `pkiserv.tmpl` file must be defined in a corresponding INSERT section.

Unlike the APPLICATION sections and TEMPLATE sections, INSERT sections can have no subsections. The format of an INSERT section is:

<INSERT NAME=*insert-name*>...</INSERT>
An INSERT contains HTML that either:

- Defines a certificate field
- Defines other common HTML that can be referenced in other sections.

The following example of an INSERT defines a certificate field.

Example:

```
<INSERT NAME=Country>
<p> <LABEL for="countryfield">Country [optfield]</LABEL> <BR>
<INPUT NAME="Country" TYPE="text" SIZE=2 maxlength="2" id="countryfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidCountry(frm){
 if ("[optfield]" == "" && frm.Country.value == "") {
  alert("Enter required field."); frm.Country.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</INSERT>
```

The next example defines other common HTML:

Example:

```
<INSERT NAME=-pagefooter>
<p>email: webmaster@your_company.com
</INSERT>
```

To reference an INSERT, you use a named field of the form

%%*insert-name*%%, for example %%Country%% or %%-pagefooter%%.

The `pkiserv.tmpl` certificate templates file contains INSERT sections of several main types:

- INSERTs that are for internal processing. (This is common HTML for web page content as listed in Table 31 on page 138.)
- INSERTs that are related to the certificate content. (See Table 32 on page 139.) These include:
  - X.509 fields (for example, `OrgUnit`)
  - Non-X.509 fields (for example `UserId`).

Table 31. INSERTs that are common HTML for web page content

INSERT NAME	Contents
-AdditionalHeadIE	ActiveX controls to enable Internet Explorer to generate a key pair.
-ChallengePassphrase, -ChallengePassPhrase2	HTML for a web page that requests the passphrase that was specified when a certificate request was submitted.
-ObjectHeaderIE	ActiveX controls to enable Internet Explorer to generate a key.
-RecoverEmail, -RecoverEmail2	HTML for the web page that requests the email address that was used when a certificate was requested, after a user requests to recover the certificate.
-requestok	HTML for the web page "Request submitted successfully" after a successful certificate request (for both original requests and renewals). (For a sample of this web page, see Figure 46 on page 389.)
-requestbad	HTML for the web page "Request was not successful".
requestor	HTML for a web page that requests the name of a certificate requestor.
requestor2	HTML for a web page that requests the email address of a certificate requestor.

Table 31. INSERTs that are common HTML for web page content (continued)

INSERT NAME	Contents
-renewkeysetIE	ActiveX controls to create a renewal certificate request using the original certificates key pair.
-renewkeysetNS	Mozilla-based browser script for renewal certificate requests.
-renewrevokeok	HTML for the web page "Request submitted successfully" after a successful attempt to revoke a certificate. (See Figure 55 on page 397 for a sample of the web page to renew or revoke a certificate.)a
-renewrevokebad	HTML for the web page "Request was not successful" after an unsuccessful attempt to renew or revoke a certificate. (See Figure 55 on page 397 for a sample of the web page to renew or revoke a certificate.)
-returnp12cert	HTML for a web page that displays a PKCS #12 package.
-preregok	HTML for the web page "Preregistration successful" after a successful attempt to preregister a client for a certificate.
-returnpkcs10	This returns the server certificate in B64 format.
returnbrowsercertNS	This contains `[base64cert]`, which is the base64 substitution variable.
returnbrowsercertIE	This contains a script for producing a popup window installing your certificate (if you are using the Microsoft Internet Explorer browser). See Figure 49 on page 392 for a sample of this web page.

Named fields in INSERT sections

Most of the following fields are X.509 fields. Table 32 on page 139 summarizes the named fields in INSERT sections. (See **Restrictions** at the end of the table.)

Table 32. Named fields in INSERT sections

Field	Description
AltDomain	The host name of the machine where a certificate is installed. This is a text field of up to 100 characters. The field can be repeated. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
AltEmail	The user's email address, including the `@` character and any periods (`.`). This is a text field of up to 100 characters. The field can be repeated. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.

Table 32. Named fields in INSERT sections (continued)

Field	Description
AltIPAddr	The unique IP address that specifies the location of the server or device on the Internet. The field can be repeated. PKI Services supports both IP version 4 and IP version 6 addresses. The IP address is a text field of up to 45 characters. • For IP version 4, the IP address is in dotted decimal format; for example, `9.67.97.103`. • For IP version 6, the IP address is divided into eight 16-bit hexadecimal blocks separated by colons. Leading zeros in each 16-bit field are optional, and successive fields of zeros can be represented by double colons, but only once; for example `1:2::3:4` is equivalent to `0001:0002:0000:0000:0000:0000:0003:0004`. • In a mixed IP version 4 and IP version 6 environment, the IP address can be expressed in the format *x:x:x:x:x:x:d.d.d.d,* where the *x* values are the hexadecimal values of the six high-order 16-bit pieces of the address, and the *d* values are the decimal values of the four low-order 8-bit pieces of the address in standard IP version 4 representation; for example, `0:0:0:0:0:0:ABCD:1.2.3.4`, or the equivalent value `::ABCD:1.2.3.4` **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
AltOther "1" on page 146	A free form value for the other name of the subject's alternate name. Unlike the other INSERTs, you must customize it before you use it. The name of this INSERT consists of the string `AltOther`, concatenated with an underscore (_), then followed by the OID, specified in the following format: `AltOther_1_2_3_4_5`. (See "Customizing the OtherName field" on page 229.) You can have more than one input field but the total length of these fields together with the length of the OID and the comma cannot exceed 255 bytes. The resulting AltOther field is built by concatenating the dotted decimal OID that matches the INSERT name, a comma, and the value of the input field. This is a text field of up to 255 characters. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
AltURI	A name or address referring to an Internet resource; a URL is one type of uniform resource identifier. This is a text field of up to 100 characters. The field can be repeated. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
BusinessCat	The business category. This is a text field of up to 64 characters. **Note:** This field is intended for use in certificates that follow the criteria for Extended Validation (EV) certificates. For more information about the criteria, see the Extended Validation Certificate Guidelines (cabforum.org/extended-validation) produced by the CA/Browser Forum.
ChallengePassPhrase "1" on page 146	The passphrase the user entered when requesting a certificate. The user types the same passphrase, exactly as entered on the request form. This is a case-sensitive text field of up to 32 characters.

Table 32. Named fields in INSERT sections (continued)

Field	Description
ClientName "1" on page 146	Name of the person or device being preregistered. This is a text field of up to 64 characters. **Restriction:** The first 32 characters of the name must be unique, irrespective of case, for each preregistered user.
CommonName	For browser certificates, this is your name, such as John Smith. (You can use your first and last name, in that order.) For server certificates, this is name by which the server's administrator wants it to be known. For SSL servers, the SSL protocol requires the CommonName to be the fully qualified domain name of the server, for example, `www.ibm.com`. CommonName is a text field of up to 64 characters. Although CommonName is a constant, no value is assigned to it. This indicates that RACF must determine the value. The user authenticates by specifying a user ID and password. (If `UserId` is listed in the APPL section, this means the application provides the user ID and password.) Providing the user ID and password enables RACF to look up the CommonName value in the user's profile. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.
Country	The country where your organization is located. This is a 2-character text field. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.
CustomExt	A custom certificate extension. Use this field to support extensions that PKI Services does not otherwise support. This is a repeatable field. For more information, see "Adding custom extensions to certificates" on page 328.
DNQualifier"1" on page 146	The subject's distinguished name qualifier. This is a text field of up to 64 characters.
DomainName"1" on page 146	The subject's domain name. It contains all the domain name components in the form <domain component1>.<domain component2>. ... .<domain component*n*>. This is a text field of up to 64 characters.
Email "1" on page 146	This is a deprecated insert for the email address for the distinguished name; use the Mail insert instead. This is a text field of up to 64 characters. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.
EmailAddr "1" on page 146	The email address for the distinguished name. This is a text field of up to 64 characters. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.

Table 32. Named fields in INSERT sections (continued)

Field	Description
ExtKeyUsage "1" on page 146	The intended purpose of the certificate. Possible values are: **clientauth** Client side authentication **cmcas** CMC archive servers **cmcca** CMC certification authorities **cmcra** CMC registration authorities **codesigning** Code signing **emailprotection** Email protection **mssmartcardlogon** Microsoft Smartcard logon **ocspsigning** OCSP response signing **pkinitclientauth** PKINIT client side authentication **pkinitkdc** PKINIT KDC **serverauth** Server side authentication **timestamping** Digital timestamping.

HostIdMap "1" on page 146	This is the user ID for authorization purposes, in an email type of format:
	`subject-id@host-name`
	For example, this could be `dsmith@ibm.com`. This is a text field of up to 100 characters.
	There are three ways to use `%%HostIdMap%%`:
	• If you place it in the CONTENT section, the end user can specify the value (or values, because it can be repeated).
	• You can also place it in the APPL section that the application provides. If you do so, it should have the following form:
	`%%HostIdMap=@host-name%%`
	The `host-name` is the hardcoded system name for the current system. The application provides the user ID as the user entered it when prompted for user ID and password. Note that, for this to function properly, the IBM HTTP Server protection scheme for the request must force a prompt for user ID and password. Thus, only one `HostIdMap` is provided using this method.
	• A third way to specify `HostIdMap` is to place `%%HostIdMap%%` in the ADMINAPPROVE section. This allows the administrator to fill in the value when approving the certificate request. See "Administering HostIdMappings extensions" on page 482 for more information.

Table 32. Named fields in INSERT sections (continued)

Field	Description
InstallCert	(This field is for the Internet Explorer browser only.) This field contains script for producing a window that installs an automatically-renewed certificate copied from an email notification.
JurCountry	The jurisdiction of incorporation country name. This is a two-character text field. **Note:** This field is intended for use in certificates that follow the criteria for Extended Validation (EV) certificates. For more information about the criteria, see the Extended Validation Certificate Guidelines (cabforum.org/extended-validation) produced by the CA/Browser Forum.
JurLocality	The jurisdiction of incorporation locality name. This is a text field of up to 64 characters. **Note:** This field is intended for use in certificates that follow the criteria for Extended Validation (EV) certificates. For more information about the criteria, see the Extended Validation Certificate Guidelines (cabforum.org/extended-validation) produced by the CA/Browser Forum.
JurStateProv	The jurisdiction of incorporation state or province name. This is a text field of up to 64 characters. **Note:** This field is intended for use in certificates that follow the criteria for Extended Validation (EV) certificates. For more information about the criteria, see the Extended Validation Certificate Guidelines (cabforum.org/extended-validation) produced by the CA/Browser Forum.
KeyProt "1" on page 146	(This field is for the Internet Explorer browser only.) This field asks if the user wants to enable strong private key protection. The drop-down choices are Yes and No.
KeySize	The size of the keys (public key and private key) in bits, if they are to be generated by PKI Services. Valid values for each key type are: **RSA** 512, 1024, 2048, 4096 **NISTECC** 192, 224, 256, 384, 521 **BPECC** 160, 192, 224, 256, 320, 384, 512
KeyUsage	The intended purpose of the certificate. Each possible value is shown in Table 33 on page 147 with its intended purpose and possible PKIX bits.
Label "2" on page 146	The label assigned to the requested certificate. This is a text field of up to 32 characters.
Locality	The city or municipality where your organization is located, such as Pittsburgh or Paris. This is a text field of up to 64 characters. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.
Mail "1" on page 146	The email address for the distinguished name. This is a text field of up to 64 characters. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.

Table 32. Named fields in INSERT sections (continued)

Field	Description
NotBefore	Number of days (0 - 30) before the certificate becomes valid.
NotAfter	Number of days (1 - 9999) that the certificate is current. For example, 365 for a one-year certificate.
NotifyEmail "1" on page 146	The email address for notification purposes. If automatic certificate renewal is in effect, this is the email address to which PKI Services sends the certificate when it is automatically renewed. This is a text field of up to 64 characters. **Notes:** 1. When a certificate is created and posted to LDAP, the NotifyEmail value, if specified, is posted as the MAIL attribute. If the MAIL attribute already exists in that directory entry, its value is replaced by the new value. If both NotifyEmail and Email appear on one request, they must have the same value. 2. If a certificate for which PKI Services generated the keys is renewed, the NotifyEmail field is ignored, and the renewed certificate is sent to the requestor's email address.
Org	Organization. The legally registered name (or trademark name, for example, IBM) of your organization. This is a text field of up to 64 characters. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.
OrgUnit	The name of your division or department. This is a text field of up to 64 characters. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.
OrgUnit2	The name of your division or department. (There can be more than one organizational unit field on a request form. For example, one could be for your department and another for your division.) This is a text field of up to 64 characters. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.
PassPhrase "1" on page 146	The user decides this and enters and then reenters it when requesting a certificate (and must later supply this value when retrieving the certificate). This is a case-sensitive text field of up to 32 characters. There is no minimum number of characters, and the user can use any characters, but alphanumeric characters (A - Z, a - z, and 0 - 9) are suggested.
PostalCode "1" on page 146	The zip code or postal code. This is a text field of up to 64 characters. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.

Table 32. Named fields in INSERT sections (continued)

Field	Description

PublicKey	The base64-encoded #10 certificate request. (This is for server or device enrollment only.) You create a certificate request on behalf of another server (which could be a z/OS server or other type of server) or device for which you are requesting a certificate. You use software specific to that server to generate the #10 request before going to the PKI Services website. Save the request in a file. Then open the file in a text editor such as Windows Notepad and copy and paste the contents into the text box on the enrollment form. A text area of 70 columns and 12 rows is allocated for this certificate request. Here is an example of the certificate request:
	
	`-----BEGIN NEW CERTIFICATE REQUEST-----`
	`MIIBiDCB8gIBADAZMRcwFQYDVQQDEw5Kb2huIFEuIFB1YmxpYzCBnzANBgkqhkiG`
	`9w0BAQEFAAOBjQAwgYkCgYEAsCT1cJHAGPqi6OjAyL+xNbt8z5ngmvqO2V0O3oYu`
	`/mEnQtRM96e+2jbmDCRo5tWVklG4OYf9ZVB5biURMJFLztfa4AVdEVtun8DH2pwc`
	`wiNIZZcC1Zym5adurUmyDk64PgiiIPMQS/t0ttG4c5U8uWSK0b1J4V4f7ps+t1aG`
	`t+cCAwEAAaAwMC4GCSqGSIb3DQEJDjEhMB8wHQYDVR0OBBYEFAlKTovBBvnFqDAO`
	`1oIhtRinwRC9MA0GCSqGSIb3DQEBBQUAA4GBAIbCVpwYvppIX3HHmpKZPNY8Snsz`
	`AJrDsgAEH51WOIRGywhqKcLLxa9htoQai6cdc8RpFVTwk6UfdCOGxMn4aFb34Tk3`
	`5WYdzOiHXg8MhHiB3EruwdWs+S7Fv3JhU3FLwU6lFLfAjbVi+35iEWQymOR6mE5W`
	`CathprmGfKRsDE5E`
	`-----END NEW CERTIFICATE REQUEST-----`

PublicKeyIE "1" on page 146	(This field is for the Internet Explorer browser only.) This is the cryptographic service provider. The user selects a value from a drop-down list (Microsoft Base Cryptographic Provider or Microsoft Enhanced Cryptographic Provider).
PublicKeyNS "1" on page 146	(This field is for Mozilla-based browsers only.) This is the key size for your public/private key pair. The user selects a value from the drop-down list. Larger keys are more secure, but they also increase the time needed for connecting to a secure session.
PublicKey2IE	(This field is for the Internet Explorer browser only.) This field is the smart card cryptographic service provider. The user selects a smart card provider from a list.
PublicKey2NS	(This field is for Mozilla-based browsers only.) This field is the keygen HTML tag. It displays a menu of key sizes from which the user must choose one. When the user clicks **submit**, a key pair of the selected size is generated.
RecoverEmail, RecoverEmail2	This field is used to recover a certificate whose keys were generated by PKI Services. It contains the email address of the requestor.
Requestor "1" on page 146	The user's name, which is used for tracking the request. This can be in any format, for example, John Smith or John. J. Smith. (This can differ from the common name, especially if the request is for a server certificate.) The value is saved with the request and issued certificate, but it is not a field in the created certificate. The default value is taken from the leftmost RDN in the subject's distinguished name, truncated to 32 characters.
Requestor2	The email address of the requestor. This field is used to request a certificate with a key pair generated by PKI Services, and to retrieve such a certificate.
Security1, Security 2, ... Security*n*	Security questions used to assist recovering a certificate whose keys were generated by PKI Services. These fields can be used by the GENCERT, REQCERT and QRECOVER exits. You can have as many of these fields as you want, but the number you have must match the number that your exits handle. The fields should be numbered in order, beginning with Security1.
SerialNumber "1" on page 146	Serial number of the subject device. This is a text field of up to 64 characters.

Table 32. Named fields in INSERT sections (continued)

Field	Description

SignWith	For PKI the component and for SAF the component and key-label used to sign this certificate, indicating the provider for certificate generation. This is a text field of up to 45 characters. It can be SAF or PKI Services, as shown in the following examples.
	Examples:
	`"SAF:CERTAUTH/Local CA Cert"`
	`"PKI:"`
	For SAF, the label of the signing certificate must be included. The first example shows the SignWith field in a SAF template. It includes the signing certificate, a CERTAUTH certificate labeled `'Local CA Cert'`.
	For PKI, it is an error to include the signing certificate. The second example shows the SignWith field in a PKI template. Notice that this contains no signing certificate.

StateProv	The state or province where your organization is located. Your registration policies determine whether you spell out the full name of the state or province or use an abbreviation. This is a text field of up to 64 characters. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.
Street "1" on page 146	The street address. This is a text field of up to 64 characters. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.
Title	Job title. This is a text field of up to 64 characters. **Note:** The value is one of the relative distinguished names that is saved in the subject's distinguished name in the certificate.
TransactionId	PKISERV web pages assign this after the user requests a certificate. When it is displayed, the user needs to record this number. This is a text field of up to 56 characters.
Uid "1" on page 146	The subject's login ID. This is a text field of up to 64 characters.
UnstructAddr "1" on page 146	Unstructured address of the subject device. This is a text field of up to 64 characters.
UnstructName "1" on page 146	Unstructured device name. This is a text field of up to 64 characters.
UserId	The owning SAF user ID. This is a text field of up to 8 characters.

Restrictions:

1. This field is applicable for only PKI certificates (certificates using the `PKI:` value in the `SignWith` field).
2. This field is applicable for only SAF certificates (certificates using the `SAF:` value in the `SignWith` field).

Table 33. KeyUsage values and their intended purpose and possible PKIX bits

KeyUsage value	**Intended purpose**	**PKIX bits**
certsign	Certificate and CRL signing	`KeyCertSign` and `cRLSign`
crlsign	CRL signing	`cRLSign`
dataencrypt, dataencipherment, or dataenciph	Data encryption	`dataEncipherment`
digitalsig or digitalsignature	Authentication	`digitalSignature`
docsign or nonrepudiation	Document signing	`nonRepudiation`
handshake	Protocol handshaking (for example, SSL)	`digitalSignature` and `keyEncipherment`
keyagree or keyagreement	Key agreement	`keyAgreement`
keycertsign	Certificate signing	`keyCertSign`
keyencrypt, keyencipherment, or keyenciph	Key transport	`keyEncipherment`

Note: If **certsign**, **crlsign**, or **keycertsign** is specified, the certificate is created with the basic constraints extension to indicate that it is a certificate authority certificate, in addition to the key usage extension.

The APPLICATION sections

The APPLICATION sections identify the application domains supported by PKI Services. The default certificate templates file (`pkiserv.tmpl`) ships with two applications sections, PKISERV (for PKI administrators) and `CUSTOMERS` (for general users).

The format of the APPLICATION sections is:

<APPLICATION NAME=*appl-name*>...</APPLICATION>
Each application section begins with an application name definition.

Examples:

`<APPLICATION NAME="PKISERV">`
This application contains support for all templates and functions.

`<APPLICATION NAME="CUSTOMERS">`
This application contains support for all templates and functions but does not include the button on the PKI Services home page that directs users to the administration page.

Each APPLICATION section can contain the subsections that are shown in Table 34 on page 147.

Table 34. Subsections of the APPLICATION sections

Section or subsection	**Contents**
CONTENT	HTML for the "PKI Services Certificate Generation Application" web page. (For a sample of this web page, see “Steps for accessing the end-user web pages” on page 376.)This subsection should contain one or more named fields (see “What are named fields?” on page 137) identifying certificate templates to use for requesting or managing certificates through this application. These template names should match the HTML selection value that is associated with them.

Table 34. Subsections of the APPLICATION sections (continued)

Section or subsection	Contents
FAILURECONTENT	This subsection contains the HTML for the web page that is displayed when the certificate request submit failed. Any named fields in this subsection are interpreted as content inserts defined by INSERT sections. For PKISERV, the INSERT sections are included as part of the HTML presented to the end user.
FINDRECOVERCONTENT	This subsection contains the information for recovering a certificate in the case where the user has forgotten the passphrase. **Note:** Only certificates whose keys were generated by PKI Services are recoverable.
RECONTENT	HTML for the web page "Renew or revoke a browser certificate". The web page displays information about a certificate so that the end user can confirm that this is the correct certificate to renew or revoke. (For a sample of this web page, see Figure 55 on page 397.)This subsection uses the substitution variable [printablecert], which contains the data that is extracted from the ICL entry. (See "What are substitution variables?" on page 136.)
RECONTENT2	This subsection is similar to the RECONTENT section except that it applies to a certificate whose key was generated by PKI Services and is to be revoked.
RECOVERCONTENT	This subsection contains HTML and JavaScript functions to recover a previously issued certificate whose key was generated by PKI Services.
RENEWEDCERT	This subsection contains HTML and JavaScript functions to install an automatically renewed certificate that is copied from an email notification if using the Internet Explorer browser.
RESUCCESSCONTENT	Contains only the named field `%%-renewrevokeok%%` (whose associated INSERT contains HTML for the web page "Request submitted successfully").
REFAILURECONTENT	Contains only the named field `%%-renewrevokebad%%` (whose associated INSERT contains HTML for the web page "Request was not successful").The web page is displayed to the end user when a renewal or revocation request is unsuccessful.
RETRIEVECONTENT2	This subsection contains the HTML to allow the end user to retrieve a recovered certificate.
RETURNCERT	This subsection contains the HTML for the web page that is displayed upon successful retrieval of a recovered certificate. This section contains the named field `%%returnp12cert%%`, which indicates a PKCS #12 format.
ADMINSCOPE	This subsection is for an administration page. It contains only the named field `%%SelectCADomain%%` to prompt the administrator to choose a domain. (For more information, see "Adding a new CA domain" on page 302.)
ADMINHEADER	This subsection contains the general installation-specific HTML content for the header of all administration web pages. See "Steps for customizing the administration web pages" on page 235 for more information.

Table 34. Subsections of the APPLICATION sections (continued)

Section or subsection	Contents
ADMINFOOTER	This subsection contains the general installation-specific HTML content for the footer of all administration web pages. See "Steps for customizing the administration web pages" on page 235 for more information.

Templates that PKI Services provides

PKI Services provides the templates to request the following certificates:

- One-year SAF server certificate
- One-year SAF browser certificate
- One-year PKI SSL browser certificate (See Figure 44 on page 386 to see a sample of this web page.)
- One-year PKI SSL S/MIME browser certificate
- One-year PKI generated key certificate
- Two-year EV SSL server certificate
- Two-year PKI browser certificate for authenticating to z/OS
- Two-year PKI Authenticode - code signing server certificate
- Two-year PKI Windows logon certificate
- Five-year PKI SSL server certificate
- *n*-year PKI browser certificate for extensions demonstration
- Five-year SCEP certificate - Preregistration
- Two-Year EST Certificate Preregistration
- Five-year PKI IPSEC server (firewall) certificate
- Five-year PKI intermediate CA server certificate

The following table describes the certificate templates that PKI Services provides:

Table 35. Certificate templates PKI Services provides

Certificate template	Description
One-year SAF server certificate	This template allows end users to request a server certificate using native SAF certificate generation facilities (rather than PKI Services certificate generation facilities). The certificate is used for handshaking only (for example, SSL). This certificate is auto-approved.
One-year SAF browser certificate	This template allows end users to request a browser certificate. SAF certificate generation facilities (rather than PKI Services certificate generation facilities) create the certificate. The requestor must input a label (see Table 32 on page 139 for descriptions of fields) because the certificate is stored in a RACF database. This certificate is auto-approved.
One-year PKI SSL browser certificate	This template allows end users to request a browser certificate that PKI Services generates. The end user enters the common name. (See Table 32 on page 139 for descriptions of fields.) This template contains an ADMINAPPROVE section. Therefore, certificates requested using this template require administrator approval before being issued. The user ID and password are not required but the passphrase is required.

Table 35. Certificate templates PKI Services provides (continued)

Certificate template	Description
One-year PKI S/MIME browser certificate	This template allows end users to request a browser certificate that PKI Services generates. This is similar to the one-year PKI SSL browser certificate except the end user selects AltEmail.
One-year PKI generated key certificate	This template allows end users to request a certificate that PKI Services generates, with a public key and private key that PKI Services generates. The user must supply a name, email address, passphrase, and key size. This template requires administrator approval. You need to assess the risk of using this template. The requestor provides the transaction ID and passphrase to retrieve the private key and the certificate. The transaction ID and the passphrase entered by the requestor can be shown on the administrator pages. A malicious administrator could retrieve the certificate and the private key and use them. You should implement measures to minimize the risk of this happening; for example, check the log record on the number of retrievals or create an exit to limit the number of retrievals.
Two-year EV SSL server certificate	This template allows end users to request a two-year extended validation server certificate.
Two-year PKI browser certificate for authenticating to z/OS	This template allows end users to request a browser certificate that PKI Services generates. This certificate is similar to the one-year PKI SSL browser certificate except that it includes the `%%HostIdMap%%` INSERT and this certificate is auto-approved. `%%HostIdMap%%` is intended as a replacement for adding (and mapping) the certificate to a RACF user ID. This template specifies `%%HostIdMap=@ host-name%%` and `%%UserId%%` in the APPL section. This template does not require administrator approval but has protection through the user ID and password. (For more information about `%%HostIdMap%%`, see the `HostIdMap` field in Table 32 on page 139.)
Two-year PKI Authenticode - code signing server certificate	This template allows end users to request that a server certificate be used to sign software that is downloaded across an untrusted medium. It also demonstrates how to define extensions for template specific certificate policies and third party-provided OCSP.
Two-year PKI Windows logon certificate	This template allows end users to request a certificate to use when logging in with a smart card to a Windows desktop as an Active Directory user. This template supports requests from both Internet Explorer and Mozilla-based browsers, and supports the following cryptographic services providers (CSPs). • Datakey • Gemplus • Infineon SICRYPT • Schlumberger Support for additional CSPs can be added when you customize the template.

Table 35. Certificate templates PKI Services provides (continued)

Certificate template	Description
Five-year PKI SSL server certificate	This template allows end users to request a server certificate that PKI Services generates. This is similar to the SAF server template except that this template contains an ADMINAPPROVE section. Therefore, certificates requested using this template require administrator approval before being issued. The user ID and password are not required but the passphrase is required.
Five-year PKI IPSEC server (firewall) certificate	This template allows end users to request a server certificate that PKI Services generates. This is similar to the five-year PKI SSL server certificate except that KeyUsage constants handshake and dataencrypt are hardcoded. Also, the end user selects AltEmail, AltIPAddr, AltURI, and AltDomain.
Five-year PKI intermediate CA server certificate	This template allows end users to request a server certificate that PKI Services generates. This is similar to the PKI SSL server template except that KeyUsage is hardcoded as certsign. Also, this certificate is auto-approved (because it runs under the user ID of the requestor, that is the person requesting this must be highly authorized). The user ID and password are required, and the units of work should run under the client's ID. In other words, the end user must be someone who can do this using RACDCERT alone, that is, must have CONTROL authority to IRR.DIGTCERT.GENCERT, and so forth. Given this requirement, the administrator need not approve this. The PassPhrase is required.
Five-year SCEP certificate - Preregistration	This template supports certificate preregistration for Simple Certificate Enrollment Protocol (SCEP) clients. The PassPhrase is required.
Two-Year EST certificate - Preregistration	This template supports certificate preregistration for Enrollment Over Secure Transport (EST)
n-year PKI browser certificate for extensions demonstration	This template creates a browser certificate that has most of its information provided by the user rather than controlled by the administrator. The certificate contains all the supported extensions.

TEMPLATE sections

TEMPLATE sections define the fields that comprise a specific certificate request. They define the certificate templates that are referenced in the APPLICATION section. The `pkiserv.tmpl` certificate templates file contains a TEMPLATE section for each of the certificates that are described in Table 35 on page 149.

Each template section begins with one or more template names.

<TEMPLATE NAME=*tmpl-name*>...</TEMPLATE NAME>
The `pkiserv.tmpl` certificate templates file that ships with PKI Services includes lines like the following example:

Example:

```
<TEMPLATE NAME=1-Year PKI SSL Browser Certificate>
<TEMPLATE NAME=PKI Browser Certificate>
<NICKNAME=1YBSSL>
```

The true name of a certificate template is its actual complete name. This is the name in the first line, `1-Year PKI SSL Browser Certificate`. However, you can refer to a single template by more than one name by using an alias. The template name in the second line, `PKI Browser Certificate`, is an alias. An alias differentiates browser from server certificates. Finally, renewing a certificate requires recalling the template name, so the template name must be stored with the certificate. The NICKNAME (or short name) serves this purpose.

Notes:

1. You can have more than one alias. (Use an additional <TEMPLATE NAME=*alias*> line for each one.)
2. The value of a NICKNAME is an 8-character string.
3. SAF certificate templates do not include nicknames.

Table 36 on page 152 shows the true name, alias, and nickname for each certificate template:

Table 36. Names, aliases, and nicknames of certificate templates.

True name	**Alias**	**Nickname**
1-Year PKI SSL Browser Certificate	PKI Browser Certificate	1YBSSL
1-Year PKI S/MIME Browser Certificate	PKI Browser Certificate	1YBSM
1-Year PKI Generated Key Certificate	PKI Key Certificate	1YKRC
2-Year EV SSL Server Certificate	PKI Server Certificate	2YEVSSL
2-Year PKI Browser Certificate For Authenticating To z/OS	PKI Browser Certificate	2YBZOS
2-Year PKI Authenticode - Code Signing Certificate	PKI Server Certificate	2YIACS
2-Year PKI Windows Logon Certificate	PKI Browser Certificate	2YBWL
5-Year PKI SSL Server Certificate	PKI Server Certificate	5YSSSL
5-Year PKI IPSEC Server (Firewall) Certificate	PKI Server Certificate	5YSIPS
5-Year PKI Intermediate CA Certificate	PKI Server Certificate	5YSCA
5-Year SCEP Certificate - Preregistration	—	5YSCEPP
2-Year EST Certificate - Preregistration	—	2YESTP
n-Year PKI Certificate for Extensions Demonstration	PKI Browser Certificate	SAMPLB
1-Year SAF Browser Certificate	SAF Browser Certificate	-
1-Year SAF Server Certificate	SAF Server Certificate	-

The AUTORENEW tag is optional. It determines whether the certificate is to be automatically renewed when it approaches expiration. This tag has the form <AUTORENEW=*value*>, where *value* can have the value `Y`, `y`, `N`, or `n`. If the AUTORENEW tag has any other value, or does not immediately follow the NICKNAME tag, PKI Services operates as if the tag is not present. The tag has the following meanings:

- AUTORENEW tag not present means that the certificate is not set up for automatic renewal.
- AUTORENEW=Y means that the certificate is enabled for automatic renewal.
- AUTORENEW=N means that the certificate is eligible for automatic renewal, but automatic renewal is disabled.

  Note: Adding the AutoRenew=Y does not enable all certificates to be AutoRenewed. It enables only the newly issued certificates after the template is updated. For more information about how AutoRenew processing should be added, see Chapter 19, "Using the administration web pages," on page 405.

See "Setting up automatic renewal of certificates" on page 323 for more information.

The SYNCHRONOUS tag is optional. It determines whether the certificate is to be synchronously created when the request does not need administrator's approval. The SYNCHRONOUS tag will be ignored if the ADMINAPPROVE tag is present.

The ADMINAPPROVE tag has the form of <SYNCHRONOUS=*value*>, where value can have the value *Y*, *y*, *N*, or *n*. If the SYNCHRONOUS tag has any other value, PKI Services operates as if the tag is not present.

The tag has the following meanings:

- The SYNCHRONOUS tag is not present or SYNCHRONOUS=N, meaning that the certificate will not be generated immediately in response to the request. Additionally, when a transaction IS is returned, the certificate has not been generated yet.
- The SYNCHRONOUS=Y means that the certificate will be generated immediately in response to the request. In orther words, when a transaction ID is returned, the certificate has been generated.

See "Setting up synchronous certificate requests" on page 325 for more information.

Example:

```
<TEMPLATE NAME=1-Year PKI SSL Browser Certificate>
<TEMPLATE NAME=PKI Browser Certificate>
<NICKNAME=1YBSSL>
<AUTORENEW=Y>
SYNCHRONOUS=Y
```

TEMPLATE sections can have the following subsections:

- CONTENT
- APPL
- CONSTANT
- ADMINAPPROVE
- SUCCESSCONTENT
- FAILURECONTENT
- RETRIEVECONTENT
- RETURNCERT
- PREREGISTER

<CONTENT>...</CONTENT>

This subsection contains the HTML to display a web page to the end user requesting a certificate of a specific type. (See Figure 44 on page 386 for a sample web page.) Field names on the certificate request (such as a text box where the user enters a value for `Common Name`) match the names of INSERT sections. The following examples show the INSERT sections corresponding to the field names %%CommonName%% and %%`Requestor (optional)`%%.

Examples:

```
<INSERT NAME=CommonName>
<p><LABEL for="commonnamefield">Common Name [optfield]</LABEL><BR>
<INPUT NAME="CommonName" TYPE="text" SIZE=64  maxlength="64"
id="commonnamefield">

<INSERT NAME=Requestor>
<p><LABEL for="requestorfield">Your name for tracking this request [optfield]</
LABEL><BR>
<INPUT NAME="Requestor" TYPE="text" SIZE=32 maxlength="32" id="requestorfield">
```

Named fields in this subsection are optional if the named field contains more that one word within the %% delimiters (as in %%`Requestor (optional)`%%). The user need not supply a value for Requestor.

<APPL>...</APPL>

This subsection identifies certificate fields for which the application itself should provide values. This subsection should contain only named fields, one per line. The only supported named fields that are allowed in this section are:

- `UserId`
- `HostIdMap`

Example:

```
<APPL>
 %%UserId%%
 %%HostIdMap=@www.ibm.com%%
<APPL>
```

<CONSTANT>...</CONSTANT>

This subsection identifies certificate fields that have a constant (hardcoded) value for everyone. This subsection should contain only named fields, one per line. The syntax for specifying the values is `%%`*`field-name=field-value`*`%%`:

Example:

```
%%KeyUsage=handshake%%
```

In addition to the named fields listed in Table 32 on page 139, you can also include the following named fields in this subsection only.

Critical

Identifies a certificate extension that is to be marked critical in the issued certificates. This name-value pair can be repeated for each extension to be marked critical. Here is the list of acceptable values for **Critical**:

- `BasicContraints` (ignored as this extension is always marked critical)
- `KeyUsage` (ignored as this extension is always marked critical)
- `ExtKeyUsage`
- `SubjectAltName, AltEmail, AltIPAddr, AltDomain, AltURI`
- `HostIdMappings, HostIdMap`
- `CertificatePolicies, CertPolicies`

Example:

```
%%Critical=ExtKeyUsage%%
```

Rules:

1. If you have specified configuration file setting `PolicyRequired=T`, then specifying `%%Critical=CertPolicies%%` is ignored. The configuration file setting PolicyCritical determines if the CertificatePolicies extension is marked critical. See "Using certificate policies" on page 284 for more information.
2. When `ExtKeyUsage` is extracted from an input PKCS #10 certificate request, the critical flag in the request is ignored. Therefore, setting `%%Critical=ExtKeyUsage%%` is the only way to get the ExtKeyUsage extension marked critical.

CertPolicies

Identifies the certificate policies that are to be included in the issued certificates. The value is a vector of numbers each representing one of the PolicyName*n* values that are specified in the **CertPolicy** section of the configuration file.

Example:

```
%%CertPolicies=3 6 10%%
```

Rule: If you specified configuration file setting `PolicyRequired=T`, then specifying `%%CertPolicies=`*`any-value`*`%%` is ignored. All issued certificates have the same certificate policies as defined in the configuration file. See "Using certificate policies" on page 284 for more information.

AuthInfoAcc

Indicates the information necessary for the AuthorityInfoAccess extension. The value specifies a two-part, comma-separated string identifying the access method (`OCSP` or `IdentrusOCSP`)and the access URL. The URL must be specified in HTTP-protocol format only. (LDAP protocol is not supported.) The name-value pair can be repeated for each value that is required in the extension.

Examples:

```
%%OCSP,URL=https://ocsp.vendor.com%%
%%IdentrusOCSP,URI=https://identrus200.identrus.com%%
%%OCSP,URI=http://www.mycompany.com/PKIServ/public-cgi/caocsp%%
```

<ADMINAPPROVE>...</ADMINAPPROVE>

This optional subsection contains the named fields that the administrator can modify when approving certificate requests. (The named fields refer to INSERT sections.) When an end user requests a certificate, the certificate request might contain fields that the end user cannot see. When approving a request, the administrator can modify:

- Fields that are present and visible to the end user in the certificate request, for example `Common Name`
- Fields that are not visible to the end user but are hardcoded (in the CONSTANT subsection) in the template, for example `Organizational unit`
- Fields that are not visible to the end user and that the PKI Services administrator can add, for example, `HostIdMappings extension` or an empty `Organizational Unit` field (these are listed in the <ADMINAPPROVE> section, and either the end user did not fill them in or they are not present on the template request form).

The presence of this section (even if empty) indicates that an administrator must approve this request. The absence of this section indicates using auto-approval.

Note: In the `pkiserv.tmpl` certificate templates file, the only certificate templates that are auto-approved are:

- One-year SAF server certificate
- One-year SAF browser certificate
- Two-year PKI browser certificate for authenticating to z/OS
- Five-year PKI intermediate CA server certificate

The following tags are allowed in the ADMINAPPROVE section:

- ADMINNUM
- AltDomain
- AltEmail
- AltIPAddr
- AltOther_*OID*
- AltURI
- AuthInfoAcc
- CertPolicies
- CommonName
- Country
- Critical

- CustomExt
- DNQualifier
- DomainName
- EmailAddr
- EndDate
- ExtKeyUsage
- HostIdMap (can repeat)
- JurCountry
- JurLocality
- JurStateProv
- KeyUsage
- Locality
- Mail
- Org
- OrgUnit (can repeat)
- PostalCode
- SerialNumber
- StartDate
- StateProv
- Street
- Title
- Uid
- UnstructName
- UnstructAddr

Note: The following fields are not modifiable and are ignored in the ADMINAPPROVE section:

- Label
- PublicKey
- Requestor
- SignWith
- UserId

(For information about fields, see Table 32 on page 139.)

Example:

```
<ADMINAPPROVE>
%%KeyUsage%%
%%CommonName%%
%%OrgUnit%%
%%Org%%
%%Country%%
%%HostIdMap%%
%%HostIdMap%%
%%HostIdMap%%
%%HostIdMap%%
<ADMINAPPROVE>
```

The ADMINNUM tag is optional. It indicates the number of PKI Services administrators that must approve certificate requests that are using this template before certificates are issued for them. This tag has the form <ADMINNUM=*value*>, where *value* is a numeric value from 1 to 32. The tag has the following meanings:

- By default, all certificate requests require approval by one PKI Services administrator. If the ADMINNUM tag is not present, all certificate requests that use this template require approval by one PKI Services administrator before issuing a certificate.
- If the ADMINNUM tag does not occur within the ADMINAPPROVE subsection, PKI Services operates as if the tag is not present.
- If the ADMINNUM value is greater than 32, a value of 32 is used.
- If the ADMINNUM value is less than one or is a non-numeric value, a value of 1 is used.

Note: A request remains in Pending Approval state until the required number of individual administrative approvals is made for the request, at which time the request changes to Approved state. If an administrator issues an Approve with Modifications on a request that is in Pending Approval state, any previously made approvals are nullified, and the number of approvals that are made for the request is reset to 1.

Example:

```
<ADMINAPPROVE>
<ADMINNUM=4>
%%KeyUsage%%
%%CommonName%%
%%OrgUnit%%
%%Org%%
%%Country%%
%%HostIdMap%%
%%HostIdMap%%
%%HostIdMap%%
%%HostIdMap%%
<ADMINAPPROVE>
```

<SUCCESSCONTENT>...</SUCCESSCONTENT>

This subsection contains the HTML to display to the end user a web page indicating that the certificate request was submitted successfully. Any named fields in this subsection are interpreted as content inserts defined by INSERT sections. For PKISERV, the INSERT sections are included as part of the HTML to display a web page to the end user.

In all of the templates included with PKI Services, <SUCCESSCONTENT> contains only the named field `%%-requestok%%`. (See “What are named fields?” on page 137 for an explanation of named fields.) This contains HTML for the web page "Request submitted successfully". (For a sample of this web page, see Figure 46 on page 389.)

<FAILURECONTENT>...</FAILURECONTENT>

This subsection contains the HTML to display to the end user a web page indicating the certificate request was not submitted successfully. Any named fields in this subsection are interpreted as content inserts defined by INSERT sections. For PKISERV, the INSERT sections are included as part of the HTML to display a web page to the end user.

In all of the templates included with PKI Services, <SUCCESSCONTENT> contains only the named field `%%-requestbad%%`. (See “What are named fields?” on page 137 for an explanation of named fields.) This contains HTML for the web page that says, "Request was not successful".

<RETRIEVECONTENT>...</RETRIEVECONTENT>

This subsection contains the HTML to display to the end user a web page to enable certificate retrieval. Any named fields in this subsection are interpreted as content inserts that the INSERT sections define. For PKISERV, the INSERT sections are included as part of the HTML presented to the end user.

For a sample of a web page this section generates, see Figure 47 on page 390. You might want to look at this web page while reading the following explanation:

In all of the templates included with PKI Services, `<RETRIEVECONTENT>` contains:

- The named field `%%-copyright%%`, which displays any copyright information. (See “What are named fields?” on page 137 for an explanation about named fields.)

- The title of the web page (This appears in the banner of your browser. Figure 47 on page 390 does not include the banner header but shows only the frame containing the content and not the browser window displaying the content.)
- A JavaScript script for processing the fields that the user enters on the web page.
- A heading that says "Retrieve Your (name of certificate)". This uses the substitution variable `[tmplname]`. (See "What are substitution variables?" on page 136 for an explanation of substitution variables.)
- Text: a heading and paragraph about bookmarking this web page.
- The named field `%%TransactionId%%` - A field where you enter your transaction ID if it is not already displayed.
- A field where you enter the passphrase you entered on the certificate request form.

<RETURNCERT>...</RETURNCERT>

This subsection contains the HTML to display a web page upon successful certificate retrieval. The formats are:

- A browser certificate that can be installed into the browser.
- A server certificate that is displayed in B64 format.
- A PKCS #12 package, which contains the private key. For a browser or server certificate, this can be a single certificate or a certificate chain, depending on the authority of the user ID that does the retrieval. For a certificate with a key pair generated by PKI Services, the PKCS #12 package contains the requested certificate and its issuer certificate.

<PREREGISTER>...</PREREGISTER>

This optional subsection indicates the creation of a preregistration record for the Enrollment over Secure Transport protocol (EST) or the Simple Certificate Enrollment Protocol (SCEP) For details on these protocols, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333 and Chapter 16, "Using Enrollment over Secure Transport (EST)," on page 341. The contents of this subsection differ depending on whether it is used for SCEP or EST request approval.

Example:

```
<PREREGISTER>
AuthenticatedClient=AutoApprove
SemiauthenticatedClient=AdminApprove
UnauthenticatedClient=Reject
SubsequentRequest=AutoApprove
RenewalRequest=AutoApprove
</PREREGISTER>
```

The ADMINNUM tag is optional. It indicates the number of PKI Services administrators that must approve certificate requests for clients that are marked as AdminApprove before certificates are issued for those clients. This tag has the form <ADMINNUM=*value*>, where *value* can be a numeric value from 1 to 32. The tag has the following meanings:

- By default, all requests for clients that are marked as AdminApprove require the approval of one PKI Services administrator. If the ADMINNUM tag is not present, all certificate requests for clients that are marked as AdminApprove require approval by one PKI Services administrator before issuing a certificate.
- If the ADMINNUM tag does not occur within the PREREGISTER subsection, PKI Services operates as if the tag is not present.
- If the ADMINNUM value is greater than 32, a value of 32 is used.
- If the ADMINNUM value is less than one or is a non-numeric value, a value of 1 is used.

Note: A request remains in Pending Approval state until the required number of individual administrative approvals is made for the request, at which time the request changes to Approved state. If an administrator issues an Approve with Modifications on a request that is in Pending

Approval state, any previously made approvals are nullified, and the number of approvals that are made for the request is reset to 1.

Example:

```
<PREREGISTER>
<ADMINNUM=4>
AuthenticatedClient=AutoApprove
SemiauthenticatedClient=AdminApprove
UnauthenticatedClient=Reject
SubsequentRequest=AutoApprove
RenewalRequest=AutoApprove
</PREREGISTER>
```

In this example:

- SCEP requests originating from authenticated clients are automatically approved, and do not require an explicit approval from a PKI Services administrator.
- SCEP requests originating from a semi-authenticated client require approvals from four PKI Services administrators before a certificate is issued.
- SCEP requests originating from any unauthenticated clients are automatically rejected without an explicit action from a PKI Services administrator.
- Requests for more certificates from previously authenticated SCEP clients, including SCEP requests to renew certificates, are automatically approved, and do not require an explicit approval from a PKI Services administrator.

When intended for use in approving EST requests, this subsection must be present, but is expected to have no contents. If the subsection does have any contents, those contents are ignored.

Example:

```
<PREREGISTER>
</PREREGISTER>
```

Summary of subsections contained in certificate templates

Table 37 on page 159 summarizes the subsections that are present in the various certificate templates in the `pkiserv.tmpl` file (as it is shipped):

Table 37. Summary of subsections in certificate templates

Certificate Templates	CONTENT	APPL	CONSTANT	ADMINAPPROVE	SUCCESSCONTENT	FAILURECONTENT	RETRIEVECONTENT	RETURNCERT	PREREGISTER
1-year PKI SSL browser	X		X	X	X	X	X	X	
1-year PKI SSL S/MIME browser	X		X	X	X	X	X	X	
1-year SAF browser	X	X	X		X	X	X	X	
1-year SAF server	X	X	X		X	X	X	X	
1-year PKI key	X		X	X	X	X	X	X	
2-year EV SSL server	X		X	X	X	X	X	X	
2-year PKI browser for z/OS	X	X	X		X	X	X	X	
2-year PKI Authenticode signing server	X		X	X	X	X	X	X	
2-year PKI Windows logon browser	X		X	X	X	X	X	X	
5-year PKI SSL server	X		X	X	X	X	X	X	
5-year PKI IPSEC server (firewall)	X		X	X	X	X	X	X	

Table 37. Summary of subsections in certificate templates (continued)

Certificate Templates	CONTENT	APPL	CONSTAN T	ADMINAP PROVE	SUCCESSC ONTENT	FAILUREC ONTENT	RETRIEVE CONTENT	RETURNC ERT	PREREGIS TER
5-year PKI intermediate CA server	X	X	X		X	X	X	X	
5-year SCEP preregistration	X		X		X	X			X
5-year EST preregistration	X		X		X	X			X
n-year PKI browser extensions demo	X	X	X	X	X	X	X	X	

Summary of fields in certificate templates

The tables in this topic summarize the fields contained in each certificate template that PKI Services provides.

Certificate templates	Fields
Fields in the PKI browser certificate templates	Table 38 on page 160
Fields in the PKI server certificate templates	Table 39 on page 162
Fields in the SAF (browser and server), SCEP, EST, and PKI generated key certificate templates	Table 40 on page 164

Table 38 on page 160, Table 39 on page 162, and Table 40 on page 164 identify each template field as one of the following:

- Required
- Optional
- Provided by the application
- Constant (supplied value is shown)
- Blank (field is not present in either the CONTENT or CONSTANT section)

Table 38. Summary of fields for PKI browser certificate templates

Field name	One-year PKI SSL browser	One-year PKI S/MIME browser	Two-year PKI browser for z/OS	Two-year PKI Windows logon certificate	*n*-year PKI browser extensions demon-stration
AltDomain					Optional
AltEmail		Required		Optional	
AltIPAddr					Optional
AltOther_*OID*				Constant"1" on page 162	Optional
AltURI					Optional
AuthInfoAcc					Constant"2" on page 162
BusinessCat					
CertPolicies					Constant: 1
ClientName					

Table 38. Summary of fields for PKI browser certificate templates (continued)

Field name	**One-year PKI SSL browser**	**One-year PKI S/MIME browser**	**Two-year PKI browser for z/OS**	**Two-year PKI Windows logon certificate**	***n*-year PKI browser extensions demon-stration**
CommonName	Required		Constant "3" on page 162	Optional	
Country					Optional
Critical					
CustomExt					Optional
DomainName					Optional
DNQualifier					Optional
EmailAddr					Optional
ExtKeyUsage	Constant: `clientauth`		Constant: `clientauth`	Constants: `clientauth` and `mssmartlogon`	Optional
HostIdMap "4" on page 162			Application provides		Optional
JurCountry					
JurLocality					
JurStateProv					
KeySize					
KeyUsage	Constant: `handshake`			Constant: `digitalSig`	Required
Label					Optional
Locality					Optional
Mail (previously called Email)					Optional
NotAfter	Constant: 365		Constant: 730		Optional
NotBefore	Constant: 0				Optional
NotifyEmail	Optional				
Org	Constant: `The Firm`				Optional
OrgUnit	Constant: `Class 1 Internet Certificate CA`				Required
OrgUnit2					Optional
PassPhrase	Required				
PostalCode					Optional
PublicKey	Browser provided "5" on page 162				

Table 38. Summary of fields for PKI browser certificate templates (continued)

Field name	One-year PKI SSL browser	One-year PKI S/MIME browser	Two-year PKI browser for z/OS	Two-year PKI Windows logon certificate	*n*-year PKI browser extensions demon-stration
PublicKey2				Browser provided "5" on page 162	
Requestor	Optional				
SerialNumber					Optional
SignWith	Constant: `PKI:`				
StateProv					Optional
Street					Optional
Title					Optional
Uid					Optional
UnstructAddr					Optional
UnstructName					Optional
UserId			Application provides		

Notes:

1. The constant value is `_1_3_6_1_4_1_311_20_2_3`.
2. The constant value is `OCSP,URL=http://ocsp.dime-o-cert.com/public-cgi/caocsp`.
3. Although CommonName is a constant, no value is assigned to it. This indicates that RACF must determine the value. The user authenticates by specifying a user ID and password. (If `UserId` is listed in the APPL section, this means the application provides the user ID and password.) Providing the user ID and password enables RACF to look up the CommonName value in the user's profile.
4. The `HostIdMap` value is formed by concatenating `UserId` with `@`*`host-name`*.
5. The PublicKey and PublicKey2 fields are coded with the *browsertype* substitution variable.

Table 39. Summary of fields for PKI server certificate templates

Field name	Two-year EV SSL server	Two-year PKI Authenticode code signing server	Five-year PKI SSL server	Five-year PKI IPSEC server (firewall)	Five-year PKI intermediate CA server
AltDomain	Optional		Optional		
AltEmail	Optional	Required	Optional		
AltIPAddr	Optional		Optional		
AltOther_*OID*					
AltURI	Optional		Optional		
AuthInfoAcc	"1" on page 164	Constant "2" on page 164			
BusinessCat	Optional				
CertPolicies		Constant: 1			
ClientName					

Table 39. Summary of fields for PKI server certificate templates (continued)

Field name	**Two-year EV SSL server**	**Two-year PKI Authenticode code signing server**	**Five-year PKI SSL server**	**Five-year PKI IPSEC server (firewall)**	**Five-year PKI intermediate CA server**
CommonName	Required	Constant: `My Company Code Signing Certificate`	Optional		
Country	Required		Optional		
Critical		Constant: `ExtKeyUsage`			
CustomExt					
DNQualifier					
DomainName					
EmailAddr					
ExtKeyUsage		Constant: `codesigning`	Constant: `serverauth`		
HostIdMap“3” on page 164					
JurCountry	Required				
JurLocality	Optional				
JurStateProv	Optional				
KeySize					
KeyUsage		Constants: `digitalsig` and `docsign`	Constant: `handshake`	Constants: `handshake` and `dataencrypt`	Constant: `certsign`
Label					
Locality	Required		Optional		
Mail (previously called Email)	Optional				
NotAfter		Constant: `730`	Constant: `1825`		
NotBefore		Constant: `0`			
NotifyEmail	Optional	Required		Optional	
Org	Required	Constant: `The Firm`	Optional		
OrgUnit	Required	Optional			
OrgUnit2			Optional		
PassPhrase	Required				
PostalCode	Optional		Optional		
PublicKey	Required				
PublicKey2					
Requestor	Optional				
SerialNumber	Required				

Table 39. Summary of fields for PKI server certificate templates (continued)

Field name	**Two-year EV SSL server**	**Two-year PKI Authenticode code signing server**	**Five-year PKI SSL server**	**Five-year PKI IPSEC server (firewall)**	**Five-year PKI intermediate CA server**
SignWith		Constant: `PKI:`			
StateProv	Required		Optional		
Street	Optional		Optional		
Title					
Uid					
UnstructAddr					
UnstructName					
UserId					Application provides

Notes:

1. The constant value is `OCSP,URL=http://ocsp.dime-o-cert.com/public-cgi/caocsp`
2. The constant value is `OCSP,URL=http://ocsp.dime-o-cert.com/public-cgi/caocsp`.
3. The `HostIdMap` value is formed by concatenating `UserId` with `@`*`host-name`*.

Table 40. Summary of fields for SAF, SCEP, EST,and PKI generated key certificate templates

Field name	**One-year SAF server**	**One-year SAF browser**	**Five-year SCEP preregistration**	**Two-year EST preregistration**	**One-year PKI generated key**
AltDomain	Optional				
AltEmail	Optional				
AltIPAddr	Optional				
AltOther_*OID*					
AltURI	Optional		Optional	Optional	
AuthInfoAcc					
BusinessCat					
CertPolicies					
ClientName			Required	Required	
CommonName	Optional	Constant“1” on page 165	Optional	Optional	Required
Country	Required	Constant: US	Optional	Optional	
Critical					
CustomExt					
EmailAddr			Optional	Optional	
ExtKeyUsage			Optional	Optional	
HostIdMap“2” on page 166			Optional		
JurCountry					
JurLocality					
JurStateProv					

Table 40. Summary of fields for SAF, SCEP, EST,and PKI generated key certificate templates (continued)

Field name	One-year SAF server	One-year SAF browser	Five-year SCEP preregistration	Two-year EST preregistration	One-year PKI generated key
KeySize					Required
KeyUsage			Optional	Optional	Constant: `handshake`
Label	Required		Optional	Optional	
Locality	Optional		Optional	Optional	
Mail (previously called Email)			Optional	Optional	
NotAfter	Constant: 365		Constant: 1825	Constant: 1825	Constant: 365
NotBefore	Constant: `0`				
NotifyEmail			Optional	Optional	
Org	Required	Constant: The `Firm`	Optional	Optional	Constant: The `Firm`
OrgUnit	Required	Constants: `OrgUnit=SAF template certificate and OrgUnit=Nuts and Bolts Division`	Optional	Optional	Constant: `Class 1 Internet Certificate CA`
OrgUnit2	Optional		Optional	Optional	
PassPhrase			Required	Required	Required
PostalCode			Optional	Optional	
PublicKey	Required"3" on page 166	Browser provided"4" on page 166	Optional	Optional	
PublicKey2					
Requestor			Optional	Optional	Required
SerialNumber			Optional	Optional	
SignWith	Constant: `SAF:CERTAUTH /taca`		Constant: `PKI:`	Constant: `PKI:`	Constant: `PKI:`
StateProv	Optional		Optional	Optional	
Street			Optional	Optional	
Title			Optional	Optional	
UnstructAddr			Optional	Optional	
UnstructName			Optional	Optional	
UserId	Application provides			Optional	

Notes:

1. Although CommonName is a constant, no value is assigned to it. This indicates that RACF must determine the value. The user authenticates by specifying a user ID and password. (If `UserId` is listed

in the APPL section, this means the application provides the user ID and password.) Providing the user ID and password enables RACF to look up the CommonName value in the user's profile.

2. The `HostIdMap` value is formed by concatenating `UserId` with `@host-name`.
3. The PublicKey is the PKCS #10 request.
4. The PublicKey field is coded with the *browsertype* substitution variable.

Examining the pkiserv.tmpl file

This topic contains excerpts from the following sections of the `pkiserv.tmpl` file. Each excerpt contains numbered pointers that describe the important tags in each section.

- "Examining the APPLICATION section" on page 166
- "Examining the TEMPLATE section" on page 179
- "Examining the INSERT section" on page 185

The `pkiserv.tmpl` file begins with a prolog section of comments explaining main sections, subsections, named fields, and substitution variables. The prolog section is followed by a DEBUG tag that you can change from the default (DEBUG=0) to DEBUG=1 to get CGI debugging information.

Examining the APPLICATION section

The APPLICATION section of the `pkiserv.tmpl` file contains two sample applications named PKISERV and CUSTOMERS.

- "Examining the PKISERV application" on page 166
- "Examining the CUSTOMERS application" on page 168

Examining the PKISERV application

The following example is an excerpt of the PKISERV application in the APPLICATION section of the `pkiserv.tmpl` file. (The vertical ellipses indicate omitted sections.)

```
# ====================================================================
#
# Application - PKISERV
#
# The installation should customize the CONTENT, ADMINHEADER
# ADMINFOOTER, and ADMINSCOPE subsections as appropriate
#
# ====================================================================
#
<APPLICATION NAME=PKISERV>  1
<CONTENT>  2
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN""http://www.w3.org/TR/html4/
loose.dtd">
<HTML lang="en"><HEAD>
#@LTM
%%-copyright%%
<TITLE>PKI Administrators Start Page</TITLE>
<!-- 17@DFD -->
%%-CheckXP%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function init()
{
  CheckXP();  //@LVA
}
//-->
</SCRIPT>
</HEAD>
<BODY onLoad="init();">
<div role="main"<>H1>PKI Administrators Start Page</H1>
<!-- 2@DFD -->
```

```
<H2>Choose one of the following:</H2>
</div>
<div role="region" aria-label="Administration Page">
<h3>Manage existing certificates and certificate requests</h3>
# The following action will force userid/pw authentication for
# administrators
<FORM name=admform METHOD=GET ACTION="/PKIServ/ssl-cgi/auth/admmain.rexx"> 3
# The following action will force client certificate authentication for
# administrators
#<FORM name=admform METHOD=GET
# ACTION="PKIServ/clientauth-cgi/auth/admmain.rexx">
<p>
<INPUT TYPE="submit" VALUE="Administration Page">
</FORM>
</div>
<div role="region" aria-label="Customers Page">
# Multiple CA mode - replicate and modify the following H3 and FORM
# section for each CA domain.
<h3>Go to the Customers' home page </h3>
<FORM name=admform METHOD=GET ACTION="/Customers/ssl-cgi/camain.rexx"> 4
#@LTD
<INPUT TYPE="submit" VALUE="Customers' Home Page">
</FORM>
</div> %%-pagefooter%%
</div>
</BODY>
</HTML>
</CONTENT>
<ADMINHEADER> 5
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN""http://www.w3.org/TR/html4/
loose.dtd">
<HTML lang="en"><HEAD>
%%-copyright%%
<TITLE>Web Based Certificate Generation Administration</TITLE>
%%-CheckXP%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function init()
{
  CheckXP();  //@LVA
}
//-->
</SCRIPT>
</HEAD>
<BODY>
</ADMINHEADER>
<ADMINFOOTER>
<p> %%-pagefooter%% 6
<BODY onLoad="init();">
</HTML>
</ADMINFOOTER>
<ADMINSCOPE> 7
# Uncomment the following line to enable multiple CA domains
#%%SelectCADomain%%
</ADMINSCOPE>
</APPLICATION>
```

The numbers in the following list refer to the highlighted tags in the preceding excerpt of the PKISERV application.

1. This is the beginning of the APPLICATION section. The name of the application is PKISERV.
2. This is the beginning of the CONTENT subsection. The CONTENT subsection contains HTML to display the web page where the administrator begins. The TITLE indicates the main heading of that web page, "PKI Administrators Start Page." (See “Steps for accessing the administration home page” on page 405 for a sample of that web page.)
3. The ACTION tag indicates where to go when the user clicks the **Administration Page** button. (See “Using the PKI Services administration home page” on page 410 for a sample of the web page.)

4. The ACTION tag indicates where to go when the user clicks the **Customers' Home Page** button. (See "Steps for accessing the end-user web pages" on page 376 for a sample of that web page.)
5. The ADMINHEADER subsection references the `%%-copyright%%` named field, which is defined in the INSERT section. This should contain the copyright statement for your company.
6. The ADMINFOOTER subsection references the `%%-pagefooter%%` named field, which is defined in the INSERT section. This named field should specify the email address of your PKI Services administrator.
7. The ADMINSCOPE subsection references the `%%SelectCADomain%%` named field, which is defined in the INSERT section. When you have multiple CA domains, you can use this variable to allow PKI administrators to select a CA domain on the administrator's home page. (See "Adding a new CA domain" on page 302 for details about implementing multiple CA domains.)

Examining the CUSTOMERS application

The following example is an excerpt of the CUSTOMERS application in the APPLICATION section of the `pkiserv.tmpl` file. (The vertical ellipses indicate omitted sections.)

```
# ====================================================================
#
# Application - CUSTOMERS
#
# The installation should customize the CONTENT subsection as appropriate.
#
# ====================================================================
#
<APPLICATION NAME=CUSTOMERS>  1 
<CONTENT>  2 
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN""http://www.w3.org/TR/html4/
loose.dtd">
<HTML lang="en"><HEAD>
#@LTM
%%-copyright%%
<TITLE> Customers Certificate Generation Application </TITLE>
<!-- @DFA -->
<SCRIPT LANGUAGE="Javascript">
<!--
//Get browser type
function getBrowserType()
{
// Determine the browser type from where the script is being invoked.
    var type = navigator.userAgent;                        <!-- @LUC -->
    if (type.indexOf("Trident")==-1 && (type.indexOf("MSIE")==-1))<!-- @LUC -->
    {
        document.getElementById('install').style.display='none';  <!-- @LUM -->
    }                                                             <!--3@LUD -->
}
// -->
</SCRIPT>
<SCRIPT LANGUAGE="JavaScript">
<!--
function getOsVersion()
{
//7@LVD
        document.getElementById('install').href  = "/PKIServ/PKICEnroll/
PKICEnrollDeploy.msi";
//@LVD
   return true;
}
//-->
</SCRIPT>
%%-CheckXP%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function init()
{
  getBrowserType();
  CheckXP();  //@LVA
```

```
}
//-->
</SCRIPT>
</HEAD>
<!-- @DFA -->
<BODY onLoad="init();">
<div role="main"><H1>PKI Services Certificate Generation Application</H1>
<div role="region" aria-label="Installations">
<p>
<A HREF="/PKIServ/cacerts/cacert.der"> 3
Install the CA certificate to enable SSL sessions for PKI Services </A>

<br><p>
<A href = "" id = "install" onClick="getOsVersion()">Install the PKI ActiveX Control to
 renew certificates</A> 4
</div>
<div role="region" aria-label="Options">
<H2>Choose one of the following:</H2>
<ul>
<li>
<div role="region" aria-label="Request A New Certificate">
<h3>Request a new certificate using a model</h3>
<FORM name=mainform METHOD=GET ACTION="/[application]/ssl-cgi/catmpl.rexx"> 5
<p><LABEL for="seltemplate">Select the certificate template to use as a
model </LABEL>
<SELECT NAME="Template" id="seltemplate"> 6
 %%1-Year PKI SSL Browser Certificate%%
     <OPTION>1-Year PKI SSL Browser Certificate
 %%1-Year PKI S/MIME Browser Certificate%%
     <OPTION>1-Year PKI S/MIME Browser Certificate
 %%2-Year PKI Windows Logon Certificate%%
     <OPTION>2-Year PKI Windows Logon Certificate
 %%2-Year PKI Browser Certificate For Authenticating To z/OS%%
     <OPTION>2-Year PKI Browser Certificate For Authenticating To z/OS
 %%5-Year PKI SSL Server Certificate%%
     <OPTION>5-Year PKI SSL Server Certificate
 %%5-Year PKI IPSEC Server (Firewall) Certificate%%
     <OPTION>5-Year PKI IPSEC Server (Firewall) Certificate
 %%5-Year PKI Intermediate CA Certificate%%
     <OPTION>5-Year PKI Intermediate CA Certificate
 %%2-Year PKI Authenticode - Code Signing Certificate%%
     <OPTION>2-Year PKI Authenticode - Code Signing Certificate
 %%5-Year SCEP Certificate - Preregistration%%
     <OPTION>5-Year SCEP Certificate - Preregistration
%%2-Year EST Certificate - Preregistration%%
    <OPTION>2-Year EST Certificate - Preregistration
 %%1-Year PKI Generated Key Certificate%%
     <OPTION>1-Year PKI Generated Key Certificate
 %%n-Year PKI Certificate for Extensions Demonstration%%
     <OPTION>n-Year PKI Certificate for Extensions Demonstration
 %%1-Year SAF Browser Certificate%%
     <OPTION>1-Year SAF Browser Certificate
 %%1-Year SAF Server Certificate%%
     <OPTION>1-Year SAF Server Certificate
 %%2-Year EV SSL Server Certificate%%
     <OPTION>2-Year EV SSL Server Certificate
</SELECT>
<p>
<INPUT TYPE="submit" VALUE="Request Certificate">
</FORM>
</div>
<li>
<div role="region" aria-label="Pick Up Previously Requested Certificates">
<h3>Pick up a previously requested certificate</h3>
<FORM name=selform METHOD=GET
   ACTION="/[application]/ssl-cgi/caretrieve.rexx" onSubmit=
   "return ValidateEntry(this)">
#-- User input fields and validation Javascript ---------------
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidateEntry(frm){
if (ValidTransactionId(frm)) {
```

```
# Add your own Javascript here if needed ---
 return true;
}
else
 return false;
}
//-->
</SCRIPT>
%%-TransactionId%%
<br>
<LABEL for="rettemplate">Select the certificate return type </LABEL>
<SELECT NAME="Template" id = "rettemplate">
 %%PKI Browser Certificate%%
     <OPTION>PKI Browser Certificate
 %%PKI Server Certificate%%
     <OPTION>PKI Server Certificate
 %%PKI Key Certificate%%
     <OPTION>PKI Key Certificate
 %%SAF Browser Certificate%%
     <OPTION>SAF Browser Certificate
 %%SAF Server Certificate%%
     <OPTION>SAF Server Certificate
</SELECT>
#-- End user input fields and validation Javascript ------------
<p>
<INPUT TYPE="submit" VALUE="Pick up Certificate">
</FORM>
</div>
<li>
<div role="region" aria-label="Renew or Revoke">
<h3>Renew or revoke a previously issued browser certificate</h3>
<FORM name=selform METHOD=GET ACTION="/[application]/clientauth-cgi/cadisplay.rexx">
<p>
<SCRIPT LANGUAGE="JavaScript">
<!--
function RenewRevokeAlert(){
var STRING_RenewRevokePrompt=
                  "You will be prompted by the browser to select " +
                  "the certificate you want to renew or revoke. " +
                  "Once you select the certificate you will be " +
                  "given the opportunity to confirm your selection. " +
                  "Note that you can only renew or revoke a single " +
                  "certificate per one browser session. If you wish " +
                  "to renew or revoke another certificate, you must " +
                  "close your browser and restart it.";
 alert(STRING_RenewRevokePrompt);
 return true;
}
//-->
</SCRIPT>
<INPUT TYPE="submit" VALUE="Renew or Revoke Certificate"
onClick="return RenewRevokeAlert()">
</FORM>
</div>
<li>
<div role="region" aria-label="Recover Certificate">
<h3>Recover a previously issued certificate whose key was generated by PKI Services </h3>
<!-- @DIC -->
<FORM name=recvform METHOD=GET
   ACTION="/[application]/ssl-cgi/carecover.rexx">
<!-- 27@DID -->
<INPUT TYPE="submit" VALUE="Recover Certificate">
</FORM>
</div>
</ul>
</div>
<p> %%-pagefooter%%
</div>
</BODY>
</HTML>
</CONTENT>
<RECONTENT>  7
```

```
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en">   <HEAD>
#@LTM
%%-copyright%%
<TITLE> Customers Renew or Revoke a Browser Certificate </TITLE>
<div role="region" class="invisible" style="font-size:0pt;" aria-label="Certificate Manager
Object">
<SCRIPT LANGUAGE="JavaScript">
<!--                                                            // @LKA
function LoadActiveX()
{
//12@LVD
//Modified code to handle alternate text for objects @LTC
   var obj = document.createElement("obj");
   obj.innerHTML = "<div role='region' aria-label='cenrollreq'>" +
   "<OBJECT classid='clsid:65D22D38-D2D2-421F-BDFE-B7D990DDFE96' id='cenrollreq'>
     cert enroll object is not available</OBJECT>" +
   "</div>";
   document.body.appendChild(obj);
//2@LVD
   return true;
}
//-->
</SCRIPT>
#@DLC
#@LVC
%%-ObjectHeader[browsertype]%%
%%-CheckXP%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function init()
{
  // 1@DLD
  LoadActiveX();
  CheckXP();  //@LVA
}
//-->
</SCRIPT>
#13@LTD
</div>
</HEAD>
<BODY onLoad="init();">
<div role="main"><H1>Renew or Revoke a Browser Certificate</H1>
<h3>Here is the certificate you selected:</h3>
<p>
[printablecert]
<h2>If this is the correct certificate, choose one of the following:</h2>
<STRONG>(otherwise you need to restart your browser to pick another certificate)
</STRONG>
</div>
# defined style sheets for displaying labels                          @LTA
<style type="text/css">
<!--
.invisible {
 height: 0px;
 width: 0px;
 overflow: hidden;
}
.invisible2 {
 visibility: hidden;
}
//-->
</style>
<div role="region" aria-label="Actions">
<ul>
<li>
<div role="region" aria-label="Renew the above certificate">
<h3>Renew the above certificate</h3>
<FORM name=renform METHOD=POST
 ACTION="/[application]/clientauth-cgi/camodify.rexx" OnSubmit=
 "return ValidateEntry(this)">
<INPUT NAME="action" TYPE="hidden" VALUE="renew">
#@LVD
<!-- @DFA -->
<INPUT NAME="autorenflag" TYPE="hidden" value =0 id="autorenflag">
#-- User input fields and validation Javascript ---------------
#-- Added call to ValidRenewKeySet to ValidateEntry function @01C
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidateEntry(frm){
if (ValidNotifyEmail(frm)
 && ValidPassPhrase(frm)
 && ValidRenewKeySet(frm)
   ) {
# Add your own Javascript here if needed ---
  return true;
 }
 else
  return false;
}
//-->
</SCRIPT>
<STRONG>*Email address for notification purposes will be ignored if
the key was generated by PKI Services</STRONG>
%%NotifyEmail (optional)%%
%%PassPhrase%%
# Add RenewKeySet for the browse type in use. @01A
%%-RenewKeySet[browsertype]%%
#-- End user input fields and validation Javascript ------------
<p>
<INPUT TYPE="submit" VALUE="Renew">
</FORM>
</div>
</li>
<li>
<div role="region" aria-label="Revoke the above certificate">
<h3>Revoke the above certificate</h3>
<FORM name=revform METHOD=POST
 ACTION="/[application]/clientauth-cgi/camodify.rexx">
<INPUT NAME="action" TYPE="hidden" VALUE="revoke">
<INPUT TYPE="submit" VALUE="Revoke">
#@LMA
#1@LRD
<span class="invisible2" style="font-size:0pt;">
<LABEL for="reasonfield">Revocation Reason</LABEL>
</span>
<SELECT NAME="reason" id="reasonfield" title="Revocation Reason">
 <OPTION Selected VALUE="0">No Reason
 <OPTION VALUE="1">User key was compromised
 <OPTION VALUE="2">CA key was compromised
 <OPTION VALUE="3">User changed affiliation
 <OPTION VALUE="4">Certificate was superseded
 <OPTION VALUE="5">Original use no longer valid
</SELECT>
</div>
</FORM>
</li>
<li>
<div role="region" aria-label="Suspend the above certificate">
<h3>Suspend the above certificate</h3>
<FORM name=suspform METHOD=POST
 ACTION="/[application]/clientauth-cgi/camodify.rexx">
<INPUT NAME="action" TYPE="hidden" VALUE="suspend">
<INPUT TYPE="submit" VALUE="Suspend">
</FORM>
</div>
</li>
</ul>
</div>
<div role="region" aria-label="Home Page">
<p>
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/camain.rexx">
```

```
<center>
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</center>
</div>
<p> %%-pagefooter%%
</BODY>
</HTML>
</RECONTENT>
<RECONTENT2> 8
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
#@LTM
%%-copyright%%
#@DLC
#@LVC
%%-ObjectHeader[browsertype]%%
%%-CheckXP%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function init()
{
  // 1@DLD
//LoadActiveX();
  CheckXP();  //@LVA
}
//-->
</SCRIPT>
#8@DLD
</HEAD>
#@DLC
<BODY>
<div role="main"><H1>Revoke a Browser Certificate</H1>
<TITLE> Customers Revoke a Browser Certificate </TITLE>
#13@LTD
<h3>Here is the certificate you selected:</h3>
<p>
[printablecert]
<h2>If this is the correct certificate, choose one of the following:</h2>
<STRONG>(otherwise you need to restart your browser to pick another certificate)
</STRONG>
<div role="region" aria-label="Actions">
<ul>
<h3><li>Revoke the above certificate</h3>
<FORM name=revform METHOD=POST
 ACTION="/[application]/clientauth-cgi/camodify.rexx">
<INPUT NAME="action" TYPE="hidden" VALUE="revoke">
<INPUT TYPE="submit" VALUE="Revoke">
<SELECT NAME="reason">
 <OPTION Selected VALUE="0">No Reason
 <OPTION VALUE="1">User key was compromised
 <OPTION VALUE="2">CA key was compromised
 <OPTION VALUE="3">User changed affiliation
 <OPTION VALUE="4">Certificate was superseded
 <OPTION VALUE="5">Original use no longer valid
</SELECT>
</FORM>
<h3><li>Suspend the above certificate</h3>
<FORM name=suspform METHOD=POST
 ACTION="/[application]/clientauth-cgi/camodify.rexx">
<INPUT NAME="action" TYPE="hidden" VALUE="suspend">
<INPUT TYPE="submit" VALUE="Suspend">
</FORM>
</ul>
</div>
<div role="region" aria-label="Home Page">
<p>
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/camain.rexx">
<center>
<INPUT TYPE="submit" VALUE="Home Page">
```

```
</FORM>
</center>
</div>
<p> %%-pagefooter%%
</div>
</BODY>
</HTML>
</RECONTENT2>
<RESUCCESSCONTENT> 9
 %%-renewrevokeok%%
</RESUCCESSCONTENT>
<REFAILURECONTENT> 10
 %%-renewrevokebad%%
</REFAILURECONTENT>
<!-- @DIA -->
<RECOVERCONTENT> 11
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
%%-copyright%%
<TITLE> Recover Certificate </TITLE>
%%-CheckXP%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function init()
{
  CheckXP();  //@LVA
}
//-->
</SCRIPT>
</HEAD>
<BODY onLoad="init();">
<div role="main"><H1>Recover previously issued certificate</H1>
<span role="region" aria-label="Recover Certificate">
<FORM name=recvform METHOD=POST
   ACTION="/[application]/ssl-cgi-bin/caqryrcvr.rexx" onSubmit=
          "return ValidateEntry(this)">
#-- User input fields and validation Javascript ---------------
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidateEntry(frm){
if (ValidRecoverEmail(frm) &&
    ValidChallengePassPhrase2(frm)){
 return true;
}
else
 return false;
}
//-->
</SCRIPT>
%%-RecoverEmail%%
%%-ChallengePassPhrase2%%
<br><br>
#Uncommented the following lines(GetSec script) if you implement security questions
#<A HREF="JavaScript:GetSec();">Click here if you forget the pass phrase</A>
#<p>
#<SCRIPT LANGUAGE="JavaScript">
#<!--
#function GetSec(){
#var addr=document.recvform.RecoverEmail.value;
#window.location.href=
#'/[application]/ssl-cgi-bin/cagorcvr.rexx?RecoverEmail='+ escape(addr);
#}
#//-->
#</SCRIPT>
<br>
<INPUT TYPE="submit" VALUE="Recover Certificate">
</FORM>
</span>
<div role="region" aria-label="Home Page">
<p>
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/camain.rexx">
```

```
<INPUT TYPE="submit" VALUE="Home Page">
</div>
<br>
<p> %%-pagefooter%%
</FORM>
</BODY>
</HTML>
<FINDRECOVERCONTENT> 12
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
%%-copyright%%
<TITLE> Use security questions to locate certificate </TITLE>
</HEAD>
<BODY>
# This ACTION forces userid/pw authentication and runs the task
# under the client's ID
#<FORM NAME=findrecoverform METHOD=POST ACTION=
#       "/[application]/ssl-cgi-bin/auth/caqryrcvr.rexx" onSubmit=
#
# This ACTION forces userid/pw authentication but runs the task
# under the surrogate ID
#<FORM NAME=findrecoverform METHOD=POST ACTION=
#       "/[application]/ssl-cgi-bin/surrogateauth/caqryrcvr.rexx" onSubmit=
#
# This ACTION is for non z/OS clients. The task runs under surrogate ID
 <FORM NAME=findrecoverform METHOD=POST ACTION=
       "/[application]/ssl-cgi-bin/caqryrcvr.rexx" onSubmit=
      "return ValidateEntry(this)">
#-- User input fields and validation Javascript ---------------
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidateEntry(frm){
if (ValidRecoverEmail(frm) &&
    ValidSecurity1(frm) &&
    ValidSecurity2(frm)) {
 return true;
}
else
 return false;
}
//-->
</SCRIPT>
<div role="main"><p><H1>Recover your certificate</H1><p></div>
<div role="region" aria-label="Recover Certificate">
<H3>Security questions - answer the following with the same answers
you provided in the original request if you forget the pass phrase.
</H3>
%%-RecoverEmail2%%
%%Security1%%
%%Security2%%
<p>
<INPUT TYPE="submit" VALUE="Recover Certificate">
</div>
</FORM>
<div role="region" aria-label="Home Page">
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/camain.rexx">
<p>
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</div>
<p>%%-pagefooter%%
</div>
</BODY>
</HTML>
<RETRIEVECONTENT2> 13
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
%%-copyright%%
```

```
<TITLE> Web Based PKIX Certificate Recovery Application</TITLE>
</HEAD>
<BODY>
<div role="main"><H1> Retrieve your recovered certificate </H1>
<div role="region" aria-label="Retrieve Certificate">
# This ACTION forces userid/pw authentication and runs the task
# under the client's ID
#<FORM NAME=recoverform METHOD=POST ACTION=
#       "/[application]/ssl-cgi-bin/auth/cagetcert2.rexx" onSubmit=
#
# This ACTION forces userid/pw authentication but runs the task
# under the surrogate ID
#<FORM NAME=recoverform METHOD=POST ACTION=
#       "/[application]/ssl-cgi-bin/surrogateauth/cagetcert2.rexx" onSubmit=
#
# This ACTION is for non z/OS clients. The task runs under surrogate ID
 <FORM NAME=recoverform METHOD=POST ACTION=
       "/[application]/ssl-cgi-bin/cagetcert2.rexx" onSubmit=
      "return ValidateEntry(this)">
#-- User input fields and validation Javascript ---------------
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidateEntry(frm){
if (ValidChallengePassPhrase2(frm)) {
 return true;
}
else
 return false;
}
//-->
</SCRIPT>
#-- End user input fields and validation Javascript -----------
<p>
<LABEL for="KeyIDfield">Key ID</LABEL>
<INPUT NAME="KeyId" SIZE=50 VALUE="[keyid]" id="KeyIDfield" readonly>
<p>
<LABEL for="Serialnumberfield">Serial number</LABEL>
<INPUT NAME="SerialNo" SIZE=16 VALUE="[serialno]" id="Serialnumberfield" readonly>
%%-ChallengePassPhrase2%%
<p>
<INPUT TYPE="submit" VALUE="Retrieve Certificate">
</FORM>
</div>
<div role="region" aria-label="Home Page">
<p>
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/camain.rexx">
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</div>
<p>%%-pagefooter%%
</div>
</BODY>
</HTML>
</RETRIEVECONTENT2>
<RETURNCERT>  14
%%returnp12cert%%
</RETURNCERT>
<FAILURECONTENT>  15
 %%-requestbad%%
</FAILURECONTENT>
<!-- @DFA -->
<RENEWEDCERT>  16
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
#@LTM
%%-copyright%%
<TITLE> Install Automatic Renewed Certificate</TITLE>
#@DLC
#@LVC
%%-ObjectHeader[browsertype]%%
```

```
%%-CheckXP%%
<SCRIPT LANGUAGE="Javascript">
<!--
//Get browser type
function getBrowserType()
{
// Determine the browser type from where the script is being invoked.
    var type = navigator.userAgent;                             // @LUC
    if (type.indexOf("Trident")!=-1 || (type.indexOf("MSIE")!=-1))  // @LUC
    {
        document.getElementById("b64cert").focus();
        LoadActiveX();
    }
    else
    {
        document.write("<HTML lang='en'><HEAD><TITLE>Auto Renew Certificate</TITLE></HEAD>");
        document.write("<BODY><div role='main'><H1>This is intended for Internet Explorer.");
        document.write("For other browser types, save the contents of the certificate <br>");
        document.write("into a file and import the certificate file. </H1></div></BODY></
HTML>");
    }
}
// -->
</SCRIPT>
<SCRIPT LANGUAGE="JavaScript">
<!--
// Load the PKIActiveX controls on the browser
function LoadActiveX()
{
//12@LVD
//Modified code to handle alternate text for objects @LTC
   var obj = document.createElement("obj");
   obj.innerHTML = "<div role='region' aria-label='cenrollreq'>" +
   "<OBJECT classid='clsid:65D22D38-D2D2-421F-BDFE-B7D990DDFE96' id='cenrollreq'>
     cert enroll object is not available</OBJECT>" +
   "</div>";
   document.body.appendChild(obj);
//2@LVD
   return true;
}
// -->
</SCRIPT>
<!-- 29@DKD -->
<SCRIPT LANGUAGE="JavaScript">
<!--
function init()
{
  // 1@DLD;
  getBrowserType();
  CheckXP();  //@LVA
}
//-->
</SCRIPT>
</HEAD>
#12@LTD
</HEAD>
<BODY onLoad="init();">
#@LVD
<INPUT NAME="autorenflag" TYPE="hidden" value=1 id="autorenflag">
#<INPUT NAME="retry" TYPE="hidden" value=0 id="retry">
<div role="main"><h1>Install Automatic Renewed Certificate</h1>
<!-- @DGC -->
<h2> Click 'Install Certificate' to install the renewed certificate you got from the email
</h2>
<TABLE>
<!-- @DGC -->
<TR><TD>
<p><LABEL for="b64cert">Base64 encoded certificate</LABEL></p>
</TD></TR>
<TR><TD>
<TEXTAREA NAME="b64cert" COLS="70"  ROWS="12"  WRAP="OFF" id="b64cert">
</TEXTAREA>
```

```
</TD></TR>
<TR><TD>
<p>
<INPUT TYPE="BUTTON" VALUE="Install Certificate" NAME="INSTALL"
onclick="InstallCertificate()">
</TD></TR></TABLE>
%%-RenewKeySetIE%%
%%InstallCert%%
<div role="region" aria-label="Home Page">
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/camain.rexx">
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</div>
%%-pagefooter%%
</BODY>
</HTML>
</RENEWEDCERT>
<ADMINHEADER>
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
<TITLE>Web Based Certificate Generation Administration</TITLE>
</HEAD>
</div>
<BODY>
</ADMINHEADER>
<ADMINFOOTER>
<p> %%-pagefooter%%
</BODY>
</HTML>
</ADMINFOOTER>
</APPLICATION>
```

The numbers in the following list refer to the highlighted tags in the preceding excerpt of the CUSTOMERS application.

1. This is the beginning of the APPLICATION section. The name of the application is CUSTOMERS.
2. This is the beginning of the CONTENT subsection. The CONTENT subsection contains HTML to display the web page where the end user requests or retrieves a certificate. The <H1> indicates the main heading of that web page, "PKI Certificate Generation Application." (See "#unique_66/unique_66_Connect_42_mainpage" on page 376 for a sample of that web page.)
3. The HREF tag is the link to install the CA certificate in the browser.
4. The HREF tag is the link to install the PKI Services ActiveX control.
5. The ACTION tag indicates where to go when the user clicks **Request certificate**.
6. The SELECT tag produces a drop-down that lists the certificate templates the user can request. (The named fields, which are bracketed with %% symbols, are the names of the certificate templates.)
7. The RECONTENT section contains the HTML to display the web page where the end user renews or revokes a certificate. The main heading on this web page is "Renew or Revoke a Browser Certificate". It includes a JavaScript function that determines which PKI Services ActiveX programs should be loaded. (See Figure 55 on page 397 for a sample of that web page.)
8. The RECONTENT2 subsection is similar to the RECONTENT section except that it applies to a certificate whose key was generated by PKI Services and is to be revoked.
9. The RESUCCESSCONTENT subsection references the `%%-renewrevokeok%%` named field, which is defined in the INSERT section. This contains HTML for the web page displayed when the user's attempt to revoke a certificate is successful. The main heading on this web page is "Request submitted successfully". (See Figure 46 on page 389 for a sample of that web page.)
10. The REFAILURECONTENT subsection references the `%%-renewrevokebad%%` named field, which is defined in the INSERT section. This contains HTML for the web page displayed when the user's attempt to renew or revoke a certificate fails. The main heading on this web page is "Request was not successful".

11. The RECOVERCONTENT subsection contains the HTML and JavaScript to input parameters required to recover a previously issued certificate whose key was generated by PKI Services.
12. The FINDRECOVERCONTENT subsection displays security questions for users to answer when they want to recover a certificate and have forgotten the passphrase.
13. The RETRIEVECONTENT2 subsection contains the HTML to allow the end user to retrieve a recovered certificate.
14. The RETURNCERT subsection contains the HTML for the web page that is displayed upon successful retrieval of a recovered certificate. This section contains the named field %%returnp12cert%%, which indicates a PKCS #12 format.
15. The FAILURECONTENT subsection contains the HTML for the web page that is displayed when the certificate request submit failed. Any named fields in this subsection are interpreted as content inserts defined by INSERT sections. For PKISERV, the INSERT sections are included as part of the HTML presented to the end user.
16. The RENEWEDCERT subsection references the `%%RenewKeySetIE%%` named field, which is defined in the INSERT section. This field contains the HTML and JavaScript functions for the web page displayed when a user clicks a link in an email notification to install an automatically renewed certificate.

Examining the TEMPLATE section

The TEMPLATE section follows the APPLICATION section and contains several sample templates. The following example is an excerpt from the TEMPLATE section of the `pkiserv.tmpl` file. (The vertical ellipses indicate omitted sections.)

```
# =====================================================================
#
# Template Name - 2-Year PKI Browser Certificate For Authenticating
#                 to z/OS  1
# Function - Creates a 2-year certificate good for authenticating to
#            z/OS. If approved, the certificate becomes valid after
#            it's requested.
#            (You may delay the valid date by specifying a non zero
#             number for the value of 'NotBefore',
#             eg. NotBefore=5. That means if the request is approved,
#             the certificate will become valid 5 days after it's
#             requested.)
#            HostidMap is formed by putting %%Userid%% and
#            %%HostIdMap=@host-name in the APPL section.
#
# 2@DHD
#
# Other than the user input fields, all other information is hard coded.
#
# User input fields:
#  Requestor - optional
#  PassPhrase - required
#  PublicKey - required (Provided by the browser itself)
#  NotifyEmail - optional
#
#  The presence of CommonName without a value tells SAF to determine
#  the CN value from the PGMRNAME field of the user's USER profile.
#  See the RACF Callable Services Guide for more information
#
#  RACF userid/password authentication : required
#  Administrator approval              : not required
#
#
# =====================================================================
#
<TEMPLATE NAME=2-Year PKI Browser Certificate For Authenticating To z/OS>  2
<TEMPLATE NAME=PKI Browser Certificate>
<NICKNAME=2YBZOS>
#<AUTORENEW=Y>
#<SYNCHRONOUS=Y>
<CONTENT>  3
```

```
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
#@LTM
%%-copyright%% 4
<TITLE> Web Based PKIX Certificate Generation Application Pg 2</TITLE> 5
%%-ObjectHeader[browsertype]%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function init()
{
  // 1@02D
  LoadCSPs();
}
//-->
</SCRIPT>
</HEAD>

<BODY onload="init();">
<div role="main"><H1>2-Year PKI Browser Certificate For Authenticating To z/OS 6 </H1>
<div role="region" aria-label="Options">
<p>
<H2>Choose one of the following:</H2>
<p>
<ul>
<li>
<div role="region" aria-label="Request A New Certificate">
<h3>Request a New Certificate</h3>
# This ACTION forces userid/pw authentication and runs the task under
# the client's ID
#<FORM NAME="CertReq" METHOD=POST ACTION= 7
#                "/[application]/ssl-cgi-bin/auth/careq.rexx" onSubmit=

# This ACTION forces userid/pw authentication but runs the task under
# the surrogate ID
 <FORM NAME="CertReq" METHOD=POST ACTION=
               "/[application]/ssl-cgi-bin/surrogateauth/careq.rexx" onSubmit=

# This ACTION is for non z/OS clients. The task runs under the
# surrogate ID
#<FORM NAME="CertReq" METHOD=POST ACTION=
#              "/[application]/ssl-cgi-bin/careq.rexx" onSubmit=
    "return ValidateEntry(this)">

<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<p> Enter values for the following field(s) 8
#-- User input fields and validation Javascript ---------------
<SCRIPT LANGUAGE="JavaScript"> 9
<!--
function ValidateEntry(frm){
  if (ValidRequestor(frm) &&
   ValidNotifyEmail(frm) &&
   ValidPassPhrase(frm) &&
   ValidPublicKey(frm)){
# Add your validation Javascript here if needed ---
 return true;
}
else
 return false;
}
//-->
</SCRIPT>
 %%Requestor (optional)%%
 %%NotifyEmail (optional)%%
 %%PassPhrase%%
 %%PublicKey[browsertype]%%
#-- End user input fields and validation Javascript -----------
<p>
<INPUT TYPE="Submit" VALUE="Submit certificate request">
<INPUT TYPE="reset" VALUE="Clear">
</FORM>
```

```
</div>
</li>
<li>
<div role="region" aria-label="Pick Up a Previously Issued Certificate">
<H3>Pick Up a Previously Issued Certificate</H3>
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/caretrieve.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Retrieve your certificate">
</FORM>
</div>
</li>
</ul>
</div>
<p>%%-pagefooter%%  10
</div>
</BODY>
</HTML>
</CONTENT>
<APPL>  11
 %%UserId%%
 %%HostIdMap=@host-name%%
</APPL>
<CONSTANT>  12
 %%CommonName=%%
 %%OrgUnit=Class 1 Internet Certificate CA%%
 %%Org=The Firm%%
 %%KeyUsage=handshake%%
 %%ExtKeyUsage=clientauth%%
 %%NotBefore=0%%
 %%NotAfter=730%%
 %%SignWith=PKI:%%
</CONSTANT>
<SUCCESSCONTENT>  13
 %%-requestok%%
</SUCCESSCONTENT>
<FAILURECONTENT>  14
 %%-requestbad%%
</FAILURECONTENT>

<RETRIEVECONTENT>  15
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
%%-copyright%%
<TITLE> Web Based PKIX Certificate Generation Application Pg 3</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
function MissingTransIdAlert(){
var STRING_MissingTransIdPrompt=
    "Enter the transaction ID assigned to the certificate.";
if(document.retrieveform.TransactionId.value==""){
 alert(STRING_MissingTransIdPrompt);
 document.retrieveform.TransactionId.focus();
 return true;
}
else {
 return false;
 }
}
//-->
</SCRIPT>
</HEAD>

<BODY>
<div role="main"><H1> Retrieve Your [tmplname]</H1>  16
<H3>Please bookmark this page</h3>
<p>Since your certificate may not have been issued yet, we recommend
that you create a bookmark to this location so that when you return to
this bookmark, the browser will display your transaction ID.
This is the easiest way to check your status.
```

```
# This ACTION forces userid/pw authentication and runs the task
# under the client's ID
#<FORM NAME=retrieveform METHOD=POST ACTION=  17
#       "/[application]/ssl-cgi-bin/auth/cagetcert.rexx" onSubmit=
#
# This ACTION forces userid/pw authentication but runs the task
# under the surrogate ID
 <FORM NAME=retrieveform METHOD=POST ACTION=
       "/[application]/ssl-cgi-bin/surrogateauth/cagetcert.rexx" onSubmit=
#
# This ACTION is for non z/OS clients. The task runs under surrogate ID
#<FORM NAME=retrieveform METHOD=POST ACTION=
#       "/[application]/ssl-cgi-bin/cagetcert.rexx" onSubmit=
      "return ValidateEntry(this)">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
#-- User input fields and validation Javascript ---------------
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidateEntry(frm){
if (ValidTransactionId(frm) &&
    ValidChallengePassPhrase(frm)) {
# Add your own Javascript here if needed
 return true;
}
else
 return false;
}
//-->
</SCRIPT>
 %%-TransactionId%%
 %%ChallengePassPhrase (optional)%%
#-- End user input fields and validation Javascript -----------
<p>
<INPUT TYPE="submit" VALUE="Retrieve and Install Certificate">
</FORM>
</div>
<div role="region" aria-label="Home Page">
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/camain.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</div>
<p>%%-pagefooter%%
</BODY>
</HTML>
</RETRIEVECONTENT>
<RETURNCERT>  18
%%returnbrowsercert[browsertype]%%
</RETURNCERT>
</TEMPLATE>
```

```
# ====================================================================
#
# Template Name - 2-Year EST Certificate - Preregistration
#
# Function - Preregister a user or device for an EST certificate
#                   and for EST certificate fulfillment
#
# User input fields:
#  ClientName
#  PassPhrase
#  SerialNumber -optional
#  UnstructAddr -optional
#
#  RACF userid/password authentication : required
#
#
# ====================================================================
#
<TEMPLATE NAME=2-Year EST Certificate - Preregistration>
<NICKNAME=2YESTP>
<CONTENT>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
%%-copyright%%
<TITLE> Web Based PKIX Certificate Generation Application Pg 2</TITLE>
%%-ObjectHeader[browsertype]%%
%%-CheckXP%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function init()
{
  CheckXP();
}
//-->
</SCRIPT>
</HEAD>
<BODY onload="init();">
<div role="main"><H1>5-Year SCEP Certificate - Preregistration</H1>
<div role="region" aria-label="Authenticate">
<p>
# This ACTION forces userid/pw authentication and runs the task under
# the user's ID. (Must be a PKI administrator.)
<FORM NAME="CertReq" METHOD=POST ACTION=
"/[application]/ssl-cgi-bin/auth/careq.rexx" onSubmit=
"return ValidateEntry(this)">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
#-- User input fields and validation Javascript ---------------
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidateEntry(frm){
if (ValidClientName(frm) &&
ValidPassPhrase(frm) &&
ValidUnstructAddr(frm) &&
ValidSerialNumber(frm)) {
# Add your validation Javascript here if needed ---
return true;
}
else
return false;
}
//-->
</SCRIPT>
<ul>
<li>
<h3>Enter values the client must provide to authenticate</h3>
# ClientName is the only required field to preregister. ClientName
# combined with PassPhrase are the minimum required to authenticate
# the client. Any other subject name field (except UnstructName
# and CommonName) or alternate name field may be specified as
# additional authenticating information.
%%ClientName%%
%%PassPhrase%%
%%SerialNumber (Optional)%%
%%UnstructAddr (Optional)%%
</li>
</ul>
#-- End user input fields and validation Javascript -----------
<p>
<INPUT TYPE="submit" VALUE="Submit preregistration request">
<INPUT TYPE="reset" VALUE="Clear">
</FORM>
</div>
<p>%%-pagefooter%%
</div>
</BODY>
</HTML>
</CONTENT>
<CONSTANT>
%%NotBefore=0%%
%%NotAfter=730%%
%%SignWith=PKI:%%
# Any subject name or alternate name fields provided as constants will
# be treated as values the client must provide to authenticate.
#%%Org=The Firm%%
</CONSTANT>
<SUCCESSCONTENT>
%%-preregok%%
</SUCCESSCONTENT>
<FAILURECONTENT>
%%-requestbad%%
</FAILURECONTENT>
<PREREGISTER>    19
</PREREGISTER>
</TEMPLATE>
```

The numbers in the following list refer to the highlighted tags in the preceding excerpt of the TEMPLATE section.

1. The template begins with a block comment identifying the template and explaining its use and fields.
2. There are three names for each certificate (except for SAF templates, which do not include nicknames). The first TEMPLATE NAME line defines the true (actual, complete) name of the certificate. The next TEMPLATE NAME line defines an alias. (This simply differentiates browser from server certificates.) The NICKNAME defines an 8-character string.

   In each template for which you want certificates to be automatically renewed, insert the AUTORENEW tag immediately following the NICKNAME tag, if it is not already there, and set it to Y.

   In each template for which you want certificates to be synchronously created, insert the SYNCHRONOUS tag before the CONTENT tag, if it is not already there, then set it to Y.
3. The CONTENT subsection contains the HTML to display a web page to the end user requesting this type of certificate. (The CGI script `catmpl.rexx` displays this content.)
4. The `%%-copyright%%` named field displays the copyright statement.
5. The title contains the heading that appears at the beginning of the browser when the web page is displayed.
6. The heading is the main heading on the web page for requesting the selected certificate.
7. The ACTION tag indicates that the CGI script that gets control when the user clicks the **Submit certificate request** button is `careq.rexx`.
8. Fields for which the user can supply input include `%%Requestor%%`, `%%PassPhrase%%`,`%%NotifyEmail%%`, and `%%PublicKey2%%`. (These fields are named fields that are defined in the INSERT section, which is shown later.) All fields not marked optional are required. `%%PublicKey2%%` contains the substitution variable, `[browsertype]`. This is replaced at run time with `IE` or `NS`, depending on the browser the user has. This is necessary because the browsers behave differently for key generation and certificates.
9. This JavaScript script provides the underlying logic for the text entry that the user must perform.
10. The `%%-pagefooter%%` named field is defined in the INSERT section (shown later). This contains the email address of the PKI Services administrator.
11. The APPL subsection indicates the fields that `careq.rexx` itself provides, in this case, `%%UserId%%` and `%%HostIdMap%%`. (These are set from the IBM HTTP Server environment variable `REMOTE_USER`.)
12. The CONSTANT subsection has hardcoded values to use, for example (for the non-SAF certificates), the signing certificate is `PKI:`.
13. The SUCCESSCONTENT subsection contains the HTML to display upon successfully requesting the certificate. It includes the `%%-requestok%%` named field. (This is defined in the INSERT section, shown in "Examining the INSERT section" on page 185. See list item "1" on page 216.)
14. The FAILURECONTENT subsection contains the HTML to display when the certificate request is unsuccessful. This subsection contains the `%%-requestbad%%` named field. (This named field is defined in the INSERT section, shown in "Examining the INSERT section" on page 185.)
15. The `-requestok` INSERT (mentioned in list item "13" on page 184) includes an ACTION that calls `caretrieve.rexx`, which displays the HTML in the RETRIEVECONTENT subsection. The first time

the web page is displayed, it includes the transaction ID associated with the certificate request. If the user leaves the web page and then returns, the transaction ID field must be filled in. Entering the transaction ID and clicking the **Continue** button calls `cagetcert.rexx`.

16. The main heading on the web page is "Retrieve Your (Name of Certificate)".
17. The ACTION is to call `cagetcert.rexx` as list item "15" on page 184 indicates.
18. The RETURNCERT subsection contains the `%%return10cert%%` named field, which is defined in an INSERT. (See list item "4" on page 216.)
19. The PREREGISTER subsection for an EST template does not specify any directives and is intentionally left empty. If any directives are specified, they are ignored.

Examining the INSERT section

The final section of the `pkiserv.tmpl` file contains several sample INSERTS. The following example is an excerpt from the INSERT section of the `pkiserv.tmpl` file. (The vertical ellipses indicate omitted sections.)

```
#
# ======================================================================
#
# Sample INSERTS
#
# ======================================================================
#  @D3C
<INSERT NAME=-AdditionalHeadIE>
#This function must be called in the init() function of your page
<SCRIPT LANGUAGE="JavaScript">
<!--
function LoadObj()
{
//12@LVD
  var obj2 = document.createElement("obj2");
  obj2.innerHTML = "<div role='region' aria-label='g_objWCF'>" +
  "<OBJECT classid='clsid:884e2049-217d-11da-b2a4-000e7bbb2b09' id='g_objWCF'>" +
  "cert enroll object is not available " +
  "</OBJECT></div>";
  document.body.appendChild(obj2);
//@LVD
  return true;
}
//-->
</SCRIPT>
#13@LTD
</INSERT>

<INSERT NAME=-requestok>  1
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<TITLE> Web Based Certificate Generation Success</TITLE>
</HEAD>
<BODY>
<div role="main"><H1> Request submitted successfully</H1>
[errorinfo]
<p> Here's your transaction ID. You will need it to retrieve your
certificate.  Press 'Continue' to retrieve the certificate.
<p> <TABLE BORDER><TR><TD>[transactionid]</TD></TR></TABLE>
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/caretrieve.rexx">  2
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT NAME="TransactionId" TYPE="hidden" VALUE="[transactionid]">
<INPUT TYPE="submit" VALUE="Continue">
</FORM>
</div>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>

#@LEA
<INSERT NAME=-requestok2>
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

```
<HTML lang="en"><HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<TITLE> Web Based Certificate Generation Success</TITLE>
</HEAD>
<BODY>
<div role="main"><H1> Request submitted successfully</H1>
<p> A link to pick up the certificate was sent to the specified
requestor's email address at [requestor].
<p>
</div>
<div role="region" aria-label="Home Page">
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/camain.rexx">
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</div>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>

<INSERT NAME=-requestbad> 3
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<TITLE> Web Based Certificate Generation Failure</TITLE>
</HEAD>
<BODY>
<div role="main"><H1> Request was not successful</H1>
<p> Please correct the problem or report the error to your Web admin
person<br>
<PRE>
[errorinfo]
</PRE>
</div>
<div role="region" aria-label="Home Page">
<p>
<FORM METHOD=GET ACTION="/[application]/sgl-cgi/camain.rexx">
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</div>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>

<INSERT NAME=-renewrevokeok>
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<TITLE> Web Based Certificate Renew/Revoke Success</TITLE>
</HEAD>
<BODY>
<div role="main"><H1> Request submitted successfully</H1>
<div role="region" aria-label="Home Page">
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/camain.rexx">
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</div>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>

<INSERT NAME=-renewrevokebad>
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<TITLE> Web Based Certificate Renew/Revoke Failure</TITLE>
</HEAD>
<BODY>
<div role="main"><H1> Request was not successful</H1>
<p> Please correct the problem or report the error to your Web admin
person<br>
<PRE>
[errorinfo]
</PRE>
```

```
</div>
<div role="region" aria-label="Home Page">
<FORM METHOD=GET ACTION="/[application]/sgl-cgi/camain.rexx">
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</div>
<p>%%-pagefooter%%
</div>
</BODY>
</HTML>
</INSERT>

<INSERT NAME=-preregok>
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<TITLE> Certificate Preregistration Success</TITLE>
</HEAD>
<BODY>
<div role="main"><H1> Preregistration successful</H1>
[errorinfo]
<p> Here's the temporary transaction ID so you may locate the
preregistration record:
<STRONG>[transactionid]</STRONG>
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/auth/admpendtid.rexx">
<INPUT NAME="domain" TYPE="hidden" VALUE="[cadomain]">
<INPUT NAME="transactionid" TYPE="hidden" VALUE="[transactionid]">
<INPUT TYPE="submit" VALUE="Examine Preregistration Record">
</FORM>
<p>
<h3>Press 'Preregister' to preregister another client
using the same template.</h3>
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/catmpl.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Preregister">
</FORM>
</div>
<div role="region" aria-label="Administration Home Page">
<FORM METHOD=GET ACTION="admmain.rexx">
<center>
<INPUT TYPE="submit" VALUE="Administration Home Page">
</FORM>
</center>
</div>
<div role="region" aria-label="Home Page">
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/camain.rexx">
<center>
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</center>
</div>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>

<INSERT NAME=-returnpkcs10cert>  4
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<TITLE> Web Based Certificate Generation Application Pg 4</TITLE>
</HEAD>
<BODY>
<div role="main"><H1> Here's your Certificate. Cut and paste it to a file</H1>
<TABLE BORDER><TR><TD>
<PRE>
[base64cert]  5
</PRE>
</TD></TR></TABLE>
<p>
</div>
<div role="region" aria-label="Home Page">
<FORM METHOD=GET ACTION="/[application]/sgl-cgi/camain.rexx">
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</div>
<p>%%-pagefooter%%
</BODY>
```

```
</HTML>
</INSERT>

<INSERT NAME=returnbrowsercertNS>
[base64cert]
</INSERT>

#@LEA
<INSERT NAME=returnp12cert>
[p12cert]
</INSERT>

<INSERT NAME=returnbrowsercertIE>
#@LMA
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<TITLE>MSIE Certificate Install</TITLE>
#29@DKD
#@DLC
#@LVC
%%-ObjectHeader[browsertype]%%
%%-CheckXP%%
<SCRIPT LANGUAGE="JavaScript">
<!--
function init()
{
  // 1@DLD
//LoadActiveX();
  CheckXP();  //@LVA
}
//-->
</SCRIPT>
#8@DLD
#12@LTD
</HEAD>
#@DLC
<BODY onLoad="init();">
#Converted VBScript to JavaScript
<SCRIPT LANGUAGE="JavaScript">
<!--
function InstallCertOnClick(){
  var pkcs7data, errmsg, rc;
  // Added for CertEnroll API processing.
  var objEnroll;
  try{
    var pkcs7data = "[iecert]";
    // CertEnroll.dll API additions follow.
    try{
      objEnroll = g_objWCF.CreateObject("X509Enrollment.CX509Enrollment");
    }catch(err){
      objEnroll = null;
    }
    if(objEnroll !== null && typeof objEnroll === 'object'){
      //CertEnroll APIs
      try{
        objEnroll.Initialize(1);  // ContextUser
      }catch(err){
        errmsg = "Error Initializing Enrollment object. " + err.description;
        alert(errmsg);
        return;
      }
      try{
        objEnroll.InstallResponse(0, pkcs7data, 1, "");
      }catch(err){
        errmsg = "Error Installing Response. " + err.description;
        alert(errmsg);
        return;
      }
      //11@LVD
    }
  }catch(err){
    errmsg = "Your new certificate failed to install. " +
     "Please ensure that you are using the same browser " +
     "that you used when making the certificate request. " +
     "Also ensure that PKI ActiveX is installed.";
    alert(errmsg);
    return;
  }
  errmsg = "Your new certificate installed successfully.";
  alert(errmsg);
  return;
  }
// -->
</SCRIPT>
<div role="main"><h1>Internet Explorer certificate install</h1>
<p> Click &quot;Install Certificate&quot; to store your new
certificate into your browser
<TABLE>
<TR> <br>
#@LTC
 <TD><INPUT TYPE="BUTTON" onclick="InstallCertOnClick()" VALUE="Install Certificate" NAME="INSTALL" >
<FORM METHOD=GET ACTION="/[application]/ssl-cgi/camain.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT TYPE="submit" VALUE="Home Page">
</FORM>
</TD>
</TR>
</TABLE>
</div>
</BODY>
</HTML>
</INSERT>
#
# ======================================================================
#
# X.509 fields (INSERTs) valid for certificate requests
#
# ======================================================================
#
<INSERT NAME=KeyUsage>  6
<div role="region" aria-label="Key Usage">
<p> <LABEL for="keyusagefield">Indicate the key usage for the
certificate [optfield] </LABEL> <BR>
<SELECT NAME="KeyUsage" MULTIPLE id="keyusagefield">
 <OPTION VALUE="handshake">Protocol handshaking e.g., SSL (digitalSignature,keyEncipherment)
 <OPTION VALUE="certsign">Certificate and CRL signing (keyCertSign, cRLSign)
 <OPTION VALUE="docsign">Document signing (nonRepudiation)
 <OPTION VALUE="dataencrypt">Data encryption (dataEncipherment)
 <OPTION VALUE="digitalsig">Authentication (digitalSignature)
 <OPTION VALUE="keyencrypt">Key Transport (keyEncipherment)
 <OPTION VALUE="keyagree">Key agreement (keyAgreement)
 <OPTION VALUE="keycertsign">Certificate signing (keyCertSign)
 <OPTION VALUE="crlsign">CRL signing (cRLSign)
</SELECT>
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidKeyUsage(frm){
 if ("[optfield]" == "" && frm.KeyUsage.value == "") {
  alert("Enter required field."); frm.KeyUsage.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=ExtKeyUsage>
<div role="region" aria-label="Extended Key Usage">
<p> <LABEL for="extkeyusagefield">Indicate the extended key usage for the
certificate [optfield] </LABEL> <BR>
<SELECT NAME="ExtKeyUsage" MULTIPLE id="extkeyusagefield">
 <OPTION VALUE="serverauth">Server side authentication (serverAuth)
 <OPTION VALUE="clientauth">Client side authentication (clientAuth)
 <OPTION VALUE="codesigning">Code signing (codeSigning)
 <OPTION VALUE="emailprotection">Email protection (emailProtection)
 <OPTION VALUE="timestamping">Digital time stamping (timeStamping)
 <OPTION VALUE="ocspsigning">OCSP response signing (OCSPSigning)
 <OPTION VALUE="mssmartcardlogon">Microsoft Smart Card Logon (msSmartCardLogon)
 <OPTION VALUE="cmcca">CMC Certification Authorities (cmcCA)
 <OPTION VALUE="cmcra">CMC Registration Authorities (cmcRA)
 <OPTION VALUE="cmcas">CMC Archive Servers (cmcAS)
 <OPTION VALUE="pkinitkdc">PKINIT KDC (pkinitKDC)
 <OPTION VALUE="pkinitclientauth">PKINIT Client side authentication (pkinitClientAuth)
</SELECT>
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidExtKeyUsage(frm){
 if ("[optfield]" == "" && frm.ExtKeyUsage.value == "") {
  alert("Enter required field."); frm.ExtKeyUsage.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>



<INSERT NAME=NotBefore>
<div role="region" aria-label="Not Before">
<p> <LABEL for="notbeforefield">Number of days after today before the
certificate becomes current [optfield]  </LABEL> <BR>
<SELECT NAME="NotBefore" id="notbeforefield">
 <OPTION> 0
 <OPTION> 30
</SELECT>
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidNotBefore(frm){
 if ("[optfield]" == "" && frm.NotBefore.value == "") {
  alert("Enter required field."); frm.NotBefore.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=NotAfter>
<div role="region" aria-label="Not After">
<p> <LABEL for="notafterfield">Length of time that the certificate is
current [optfield] </LABEL> <BR>
<SELECT NAME="NotAfter" id="notafterfield">
 <OPTION value="365">1 Year
 <OPTION value="730">2 Years
</SELECT>
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidNotAfter(frm){
 if ("[optfield]" == "" && frm.NotAfter.value == "") {
  alert("Enter required field."); frm.NotAfter.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=Country>
<div role="region" aria-label="Country">
<p> <LABEL for="countryfield">Country [optfield]</LABEL> <BR>
<INPUT NAME="Country" TYPE="text" SIZE=2 maxlength="2"
id="countryfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidCountry(frm){
 if ("[optfield]" == "" && frm.Country.value == "") {
  alert("Enter required field."); frm.Country.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=Org>
<div role="region" aria-label="Organization">
<p> <LABEL for="orgfield">Organization [optfield]</LABEL> <BR>
<INPUT NAME="Org" TYPE="text" SIZE=64  maxlength="64" id="orgfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidOrg(frm){
 if ("[optfield]" == "" && frm.Org.value == "") {
  alert("Enter required field."); frm.Org.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

# OrgUnit is a repeatable field. If more than one is needed, a
# separate INSERT, which can be modelled from this one, is needed.
# See INSERT NAME=OrgUnit2 for an example.
<INSERT NAME=OrgUnit>
<div role="region" aria-label="Organizational Unit">
<p> <LABEL for="orgunitfield">Organizational Unit [optfield]
</LABEL> <BR>
<INPUT NAME="OrgUnit" TYPE="text" SIZE=64  maxlength="64"
id="orgunitfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidOrgUnit(frm){
 if ("[optfield]" == "" && frm.OrgUnit.value == "") {
  alert("Enter required field."); frm.OrgUnit.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=OrgUnit2>
<div role="region" aria-label="Organizational Unit 2">
<p> <LABEL for="orgunit2field">Organizational Unit [optfield]
</LABEL> <BR>
<INPUT NAME="OrgUnit2" TYPE="text" SIZE=64  maxlength="64"
id="orgunit2field">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidOrgUnit2(frm){
 if ("[optfield]" == "" && frm.OrgUnit2.value == "") {
  alert("Enter required field."); frm.OrgUnit2.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=Locality>
<div role="region" aria-label="Locality">
<p> <LABEL for="localityfield">Locality [optfield]</LABEL> <BR>
<INPUT NAME="Locality" TYPE="text" SIZE=64  maxlength="64"
id="localityfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidLocality(frm){
 if ("[optfield]" == "" && frm.Locality.value == "") {
  alert("Enter required field."); frm.Locality.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=StateProv>
<div role="region" aria-label="State or Province">
<p> <LABEL for="stateprovfield">State or Province [optfield]
</LABEL> <BR>
<INPUT NAME="StateProv" TYPE="text" SIZE=64  maxlength="64"
id="stateprovfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidStateProv(frm){
 if ("[optfield]" == "" && frm.StateProv.value == "") {
  alert("Enter required field."); frm.StateProv.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=CommonName>
<div role="region" aria-label="Common Name">
<p> <LABEL for="commonnamefield">Common Name [optfield] </LABEL> <BR>
<INPUT NAME="CommonName" TYPE="text" SIZE=64  maxlength="64"
id="commonnamefield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidCommonName(frm){
 if ("[optfield]" == "" && frm.CommonName.value == "") {
  alert("Enter required field."); frm.CommonName.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=Title>
<div role="region" aria-label="Title">
<p> <LABEL for="titlefield">Title [optfield] </LABEL> <BR>
<INPUT NAME="Title" TYPE="text" SIZE=64  maxlength="64"
id="titlefield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidTitle(frm){
 if ("[optfield]" == "" && frm.Title.value == "") {
  alert("Enter required field."); frm.Title.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=DNQualifier>
<div role="region" aria-label="Distinguished Name Qualifier">
<p> <LABEL for="dnqualfield">Distinguished Name Qualifier
[optfield] </LABEL> <BR>
<INPUT NAME="DNQualifier" TYPE="text" SIZE=64  maxlength="64"
id="dnqualfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidDNQualifier(frm){
 if ("[optfield]" == "" && frm.DNQualifier.value == "") {
  alert("Enter required field."); frm.DNQualifier.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=DomainName>
<div role="region" aria-label="Domain Name">
<p> <LABEL for="domainnamefield">Domain Name [optfield] </LABEL> <BR>
<INPUT NAME="DomainName" TYPE="text" SIZE=64  maxlength="64"
id="domainnamefield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidDomainName(frm){
 if ("[optfield]" == "" && frm.DomainName.value == "") {
  alert("Enter required field."); frm.DomainName.focus();
  return false;
 }
 return true;
}
//-->
```

```
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=Uid>
<div role="region" aria-label="User Login">
<p> <LABEL for="uidfield">User Login ID [optfield] </LABEL> <BR>
<INPUT NAME="Uid" TYPE="text" SIZE=64  maxlength="64" id="uidfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidUid(frm){
 if ("[optfield]" == "" && frm.Uid.value == "") {
  alert("Enter required field."); frm.Uid.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

# AltIPAddr, AltEmail, AltURI and AltDomain are repeatable fields. If
# more than one is needed, a separate INSERT, which can be modelled
# from the original one, is needed.
# See INSERT NAME=AltDomain2 for an example.

# Updated Size and maxlength of the AltIPAddr field to 45 to allow
# for IPv6 addresses and updated field description text
<INSERT NAME=AltIPAddr>
<div role="region" aria-label="Alternate IP Address">
<p> <LABEL for="altipaddrfield">IP address for alternate name in IPv4
or IPv6 format [optfield] </LABEL> <BR>
<INPUT NAME="AltIPAddr" TYPE="text" SIZE=45  maxlength="45"
id="altipaddrfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidAltIPAddr(frm){
 if ("[optfield]" == "" && frm.AltIPAddr.value == "") {
  alert("Enter required field."); frm.AltIPAddr.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=AltEmail>
<div role="region" aria-label="Alternate Email">
<p> <LABEL for="altemailfield">Email address for alternate name
[optfield] </LABEL> <BR>
<INPUT NAME="AltEmail" TYPE="text" SIZE=100  maxlength="100"
id="altemailfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidAltEmail(frm){
 if ("[optfield]" == "" && frm.AltEmail.value == "") {
  alert("Enter required field."); frm.AltEmail.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=AltURI>
<div role="region" aria-label="Alternate Uniform Resource Identifier">
<p> <LABEL for="alturifield">Uniform Resource Identifier for alternate
name [optfield] </LABEL> <BR>
<INPUT NAME="AltURI" TYPE="text" SIZE=100  maxlength="255"
id="alturifield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidAltURI(frm){
 if ("[optfield]" == "" && frm.AltURI.value == "") {
  alert("Enter required field."); frm.AltURI.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=AltDomain>
<div role="region" aria-label="Alternate Domain">
<p> <LABEL for="altdomainfield">Domain name for alternate name
[optfield] </LABEL> <BR>
<INPUT NAME="AltDomain" TYPE="text" SIZE=100  maxlength="100"
id="altdomainfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidAltDomain(frm){
 if ("[optfield]" == "" && frm.AltDomain.value == "") {
  alert("Enter required field."); frm.AltDomain.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=AltDomain2>
<div role="region" aria-label="Alternate Domain">
<p> <LABEL for="altdomain2field">Domain name for alternate name
[optfield] </LABEL> <BR>
<INPUT NAME="AltDomain2" TYPE="text" SIZE=100  maxlength="100"
id="altdomain2field">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidAltDomain2(frm){
 if ("[optfield]" == "" && frm.AltDomain2.value == "") {
  alert("Enter required field."); frm.AltDomain2.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=Street>
<div role="region" aria-label="Street Address">
<p> <LABEL for="streetfield">Street address [optfield] </LABEL> <BR>
<INPUT NAME="Street" TYPE="text" MAXLENGTH=64 SIZE=64
id="streetfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidStreet(frm){
 if ("[optfield]" == "" && frm.Street.value == "") {
  alert("Enter required field."); frm.Street.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=PostalCode>
<div role="region" aria-label="Postal Code">
<p> <LABEL for="postalcodefield">Zipcode or postal code [optfield]
</LABEL> <BR>
<INPUT NAME="PostalCode" TYPE="text" MAXLENGTH=64 SIZE=64
id="postalcodefield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidPostalCode(frm){
 if ("[optfield]" == "" && frm.PostalCode.value == "") {
  alert("Enter required field."); frm.PostalCode.focus();
  return false;
 }
 return true;
}
//-->
```

```
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=Email>
<div role="region" aria-label="Email Address">
<p> <LABEL for="emailfield">Email address for distinguished name
MAIL= attribute [optfield] </LABEL> <BR>
# Deprecated, use the MAIL INSERT instead
<INPUT NAME="Email" TYPE="text" MAXLENGTH=64 SIZE=64 id="emailfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidEmail(frm){
 if ("[optfield]" == "" && frm.Email.value == "") {
  alert("Enter required field."); frm.Email.focus();
  return false;
 }
 if (frm.Email.value != "") {
  for (i=0;i<frm.length;i++) {
   var fld= frm.elements[i];
   if (fld.name == "NotifyEmail")
    if (fld.value != "" && fld.value != frm.Email.value) {
     alert("Notification email cannot differ from distinguished name email.");
     frm.NotifyEmail.focus();
     return false;
    }
  }
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=Mail>
<div role="region" aria-label="Mail">
<p> <LABEL for="mailfield">Email address for distinguished name
MAIL= attribute [optfield] </LABEL> <BR>
# attribute defined in RFC2798, OID 0.9.2342.19200300.100.1.3
<INPUT NAME="Mail" TYPE="text" MAXLENGTH=64 SIZE=64 id="mailfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidMail(frm){
 if ("[optfield]" == "" && frm.Mail.value == "") {
  alert("Enter required field."); frm.Mail.focus();
  return false;
 }
 if (frm.Mail.value != "") {
  for (i=0;i<frm.length;i++) {
   var fld= frm.elements[i];
   if (fld.name == "NotifyEmail")
    if (fld.value != "" && fld.value != frm.Mail.value) {
     alert("Notification email cannot differ from distinguished name MAIL=attribute.");
     frm.NotifyEmail.focus();
     return false;
    }
  }
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=EmailAddr>
<div role="region" aria-label="Email">
<p> <LABEL for="emailaddrfield">Email address for distinguished name
EMAIL= attribute [optfield] </LABEL> <BR>
# attribute defined in RFC2798, OID 1.2.840.113549.1.9.1
<INPUT NAME="EmailAddr" TYPE="text" MAXLENGTH=64 SIZE=64
id="emailaddrfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidEmailAddr(frm){
 if ("[optfield]" == "" && frm.EmailAddr.value == "") {
  alert("Enter required field."); frm.EmailAddr.focus();
  return false;
 }
 return true;
}
```

```
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=SignWith>
<div role="region" aria-label="Sign With">
<p> <LABEL for="signwithfield">Component:/key-Label used to sign this
certificate [optfield] </LABEL> <BR>
<p> e.g., "SAF:CERTAUTH/Local CA Cert" sign by CERTAUTH certificate
 "Local CA Cert"
<INPUT NAME="SignWith" TYPE="text" SIZE=45  maxlength="45"
id="signwithfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidSignWith(frm){
 if ("[optfield]" == "" && frm.SignWith.value == "") {
  alert("Enter required field."); frm.SignWith.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=PublicKey>
<div role="region" aria-label="Public Key">
<p> <LABEL for="publickeyfield">Base64 encoded PKCS#10 certificate
request [optfield] </LABEL> <BR>
<TEXTAREA NAME="PublicKey"
  COLS="70"
  ROWS="12"
  WRAP="OFF" id="publickeyfield">
</TEXTAREA>
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidPublicKey(frm){
 if ("[optfield]" == "" && frm.PublicKey.value == "") {
  alert("Enter required field."); frm.PublicKey.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>


<INSERT NAME=PublicKeyNS>
<div role="region" aria-label="Select Key Size">
<p> <LABEL for="keygentag">Select a key size</LABEL>
<KEYGEN NAME="PublicKey" id="keygentag">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidPublicKey(frm){
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=PublicKeyIE>
<div role="region" aria-label="Public Key">
#Converted VBScript to JavaScript
<SCRIPT LANGUAGE="JavaScript">
<!--
function SendReq(){
  var pkcs10data,DN,i,Message,CommonName;
  var objEnroll;
  DN= "";
  CommonName= "Unspecified Distinguished Name";
  DN= "CN=" + CommonName + ";";
  pkcs10data = "";
  // CertEnroll APIs for enrollment processing.
  try{
    objEnroll = g_objWCF.CreateObject("X509Enrollment.CX509Enrollment");
  }catch(err){
    objEnroll = null;
```

```
   }
   if(objEnroll !== null && typeof objEnroll === 'object'){
     var objPrivateKey;
     var objRequest;
     var provider;
     var selectedCSP;
     var objCSPs;
     var objName;
     var objHash;
     try{
       objPrivateKey = g_objWCF.CreateObject("X509Enrollment.CX509PrivateKey");
     }catch(err){
       Message = "Error creating Private Key object: \n" + err.description;
       alert(Message);
       return;
     }
     try{
       objRequest = g_objWCF.CreateObject("X509Enrollment.CX509CertificateRequestPkcs10");
     }catch(err){
       Message = "Error creating Request object: \n" + err.description;
       alert(Message);
       return;
     }
     //Setup Private key properties based on the selected provider
     i = document.getElementById("cspfield").options.selectedIndex;
     provider =  document.getElementById("cspfield").options(i).text.toLowerCase();

     if((provider.indexOf("smart") > 0) || (provider.indexOf("card") > 0)){
       //For Smart Card Providers, retrieve the index of the selected CSP
       //and set the Private key name, type, and KeySpec
       objPrivateKey.ProviderName = document.getElementById("cspfield").options(i).text;
       objPrivateKey.ProviderType = document.getElementById("cspfield").options(i).value;
       objPrivateKey.KeySpec      = 1; // XCN_AT_KEYEXCHANGE
       objPrivateKey.Length       = 2048; // KeySize = 2048
     }else{
       try{
         selectedCSP = g_objWCF.CreateObject("X509Enrollment.CCspInformation");
       }catch(err){
         Message = "Error creating the a CSP Information object: /n" + err.description;
         alert(Message);
         return;
       }
       try{
         objCSPs = g_objWCF.CreateObject("X509Enrollment.CCspInformations");
       }catch(err){
         Message = "Error creating the CSP Informations object: \n" + err.description;
         alert(Message);
         return;
       }

       //Retrieve the index of the selected CSP and initialize the
       //CSPInformation object using the provider name
       selectedCSP.InitializeFromName( document.getElementById("cspfield").options(i).text );

       //Add the CSPInformation object to the CSPInformations object
       objCSPs.Add( selectedCSP );

       //Set the PrivateKey objects CspInformations to our object
       objPrivateKey.CspInformations = objCSPs;

       //Set intended usage of private key for KeyExchange purposes
       objPrivateKey.KeySpec = 1; // XCN_AT_KEYEXCHANGE
       objPrivateKey.Length  = 2048; // KeySize = 2048

       //Set KeyProtection based on user input
       if(document.CertReq.KeyProt.value == 1){
         objPrivateKey.KeyProtection = 2;
         //XCN_NCRYPT_UI_FORCE_HIGH_PROTECTION_FLAG
       }else{
         objPrivateKey.KeyProtection = 0;
         //XCN_NCRYPT_UI_NO_PROTECTION_FLAG
       }
       //==============================================================
       // The ExportPolicy is set to allow the private key to be exported,
       // other options allow the private key to be exported only once for
       // archival in a variety of formats, or prevents export of the
       // private key.
       //   ExportPolicy = 0 = XCN_NCRYPT_ALLOW_EXPORT_NONE
       //   ExportPolicy = 1 = XCN_NCRYPT_ALLOW_EXPORT_FLAG
       //   ExportPolicy = 2 = XCN_NCRYPT_ALLOW_PLAINTEXT_EXPORT_FLAG
       //   ExportPolicy = 4 = XCN_NCRYPT_ALLOW_ARCHIVING_FLAG
       //   ExportPolicy = 8 = XCN_NCRYPT_ALLOW_PLAINTEXT_ARCHIVING_FLAG
```

```
      //=================================================================
      objPrivateKey.ExportPolicy = 1; // XCN_NCRYPT_ALLOW_EXPORT_FLAG
    }
    try{
      objRequest.InitializeFromPrivateKey( 1, objPrivateKey, "");
    }catch(err){
      Message = "Error initializing request from private key " + err.description;
      alert(Message);
      return;
    }
    try{
      objName = g_objWCF.CreateObject("X509Enrollment.CX500DistinguishedName");
    }catch(err){
      Message = "Error creating X500DistinguishedName object: \n" + err.description;
      alert(Message);
      return;
    }
    try{
      objName.Encode(DN);
    }catch(err){
      Message = "Error encoding the subject distinguished name \n" + err.description;
      alert(Message);
      return;
    }
    try{
      objRequest.Subject = objName;
    }catch(err){
      Message = "Error setting the subject name in request " + err.description;
      alert(Message);
      return;
    }
    //setting signature algorithm SHA256 in request
    try {                                                        //
      objHash = g_objWCF.CreateObject("X509Enrollment.CObjectId");
      objHash.InitializeFromAlgorithmName( 4           //XCN_CRYPT_SIGN_ALG_OID_GROUP_ID
                                         , 0x00000000  //XCN_CRYPT_OID_INFO_PUBKEY_ANY
                                         , 0x00000000  //AlgorithmFlagsNone
                                         , "SHA256");            //
      objRequest.HashAlgorithm = objHash;                        //
    } catch(err) {
      Message = "Error setting the signature algorithm in request "
                + err.description;                               //
      alert(Message);                                            //
      return;                                                    //
    }                                                            //

    try{
      objEnroll.InitializeFromRequest( objRequest )
    }catch(err){
      Message = "Error initializing Enrollment object from request: " + err.description;
      alert(Message);
      return;
    }
    pkcs10data = objEnroll.CreateRequest(1); // XCN_CRYPT_STRING_BASE64
    //23@LVD
  }
  document.CertReq.PublicKey.value = pkcs10data;
  if(pkcs10data.length <= 0){
    alert("PKCS10 Creation Failed");
  }
}
//-->
</SCRIPT>

<p> Select the following key information
#
# Changed VBScript to JavaScript
<SCRIPT LANGUAGE="JavaScript">
<!--
function LoadCSPs(){
  try {
    var i;
    var csp;
    var sv;
    //Modifications for CertEnroll API enrollment process.
    var objCSPs;
    var oOption;
    var errmsg;

    try{
      objCSPs = g_objWCF.CreateObject("X509Enrollment.CCspInformations");
    }catch(err){
      objCSPs = null;
      alert("Failed to load CSPs"); //
      return;                       //
    }
    if(objCSPs !== null && typeof objCSPs === 'object'){
      //CertEnroll APIs
      objCSPs.AddAvailableCsps();
      for(i=0 ; i < objCSPs.Count;i++){
        //Only include Legacy(Crypto API) providers at this time
        if(objCSPs.ItemByIndex(i).LegacyCsp){
          oOption = document.createElement("OPTION");
          oOption.text = objCSPs.ItemByIndex(i).Name;
          oOption.value = objCSPs.ItemByIndex(i).Type;
          document.getElementById("cspfield").add(oOption);
        }
      }
      //30@LVD
    }
  }catch (e) {
    //handle
    alert("Failed to load CSPs");
  }
}
//-->
</SCRIPT>
<p><LABEL for="cspfield">Cryptographic Service Provider </LABEL>
<select name="CSP" id="cspfield">
</select>


<p> <LABEL for="strongprotfield">Enable strong private key
protection? </LABEL>
<select name="KeyProt" id="strongprotfield">
 <option value="1">Yes</option>
 <option value="0" selected>No</option>
</select>
<input type="hidden" name="PublicKey" value="">
<p>


<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidPublicKey(frm){
 SendReq();
 if (document.CertReq.PublicKey.value == "")
  return false;
 else
  return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

#
#-- Added new RenewKeySetNS insert which implements the
#-- ValidRenewKeySet function for Netscape/Mozilla based
#-- browsers. Just returns true
#
<INSERT NAME=-RenewKeySetNS>
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidRenewKeySet(frm){
 return true;
}
//-->
</SCRIPT>
</INSERT>

#
#-- Added new RenewKeySetIE insert which implements the
#-- ValidRenewKeySet function for MSIE browsers.
#

<INSERT NAME=-RenewKeySetIE>
<!--Removed CAPICOM support                                  158@LUD -->
<SCRIPT LANGUAGE="JavaScript">
<!--
function ActiveXRenewKeySet()
{                                                                    //
   var flag = document.getElementById("autorenflag").value;          //
   if(flag == 0)                                                     //
```

```
      var b64cert = "[iecert]";                                 //
   else                                                         //
      var b64cert = document.getElementById("b64cert").value;   //
   var certlen = b64cert.length;
   var OS = navigator.userAgent;
   var flag2 = false;
//24@LVD
   myax = document.getElementById("cenrollreq");
   if(myax && myax.object)
   {
     flag2 = true;
   }
   else
   {
     return 1;
   }
   if(flag2 == true)
   {
      try {
         cenrollreq.CreateCEnrollRequest(b64cert,certlen);
         return 0;
      }
      catch(e) {
        alert("PKI ActiveX failed\n" + e.description + "\nContact your PKI administrator");
        return 1;
      }
//@LVD
   }

}
-->
</SCRIPT>
<input type="hidden" name="PublicKey" value="">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidRenewKeySet(frm){
//                                                                  4D@LKA
  // -- The ValidRenewKeySet function has been modified to call the
  //--  ActiveXRenewKeySet function for MS IE browsers invokes.
  //--  On Failure the RenewKeySet function is invoked and handles
  //-- the different results based on the user's responses and
  //-- capabilities of the user's machine.
     var flag = document.getElementById("autorenflag").value;       // @DFA
     var res = ActiveXRenewKeySet();                                // @LKA
     if(res == 0)
        return true;
     else
     {
        // PKI ActiveX failed
//7@LVD
        var confirmstr = "Click OK to install PKICEnroll ActiveX Control to renew
                          certificates or Cancel to cancel the renew.";
//@LVD
        var result = confirm(confirmstr);
        if(result == true)
        {
//6@LVD
             window.location = "/PKIServ/PKICEnroll/PKICEnrollDeploy.msi";
//@LVD
        }
        else
        {
             alert("PKI ActiveX Control has to be used to renew and install certificates");
             return false;
        }
                                                               // 4@LUD
     }
}
//-->
</SCRIPT>
</INSERT>

# ======================================================================
#
# X.509 fields (INSERTs) that require customization before being used
#
# ======================================================================
# ======================================================================
# INSERT NAME=AltOther_<OID>
# Here it shows two 'AltOther' INSERTs. You may add as many as you need.
# The name of this INSERT is built with the string 'AltOther_',
# concatenated with an underscore(_) separated OID that you need.
# You may have more than one input fields. But the total length of the
# fields together with the length of the OID and the comma can not exceed
# 255.
# The result AltOther field is built by concatenating the dot(.) separated
# OID, which matches this INSERT name, a comma, and the value(s) of the
# input field(s).
# Eg., in AltOther_1_2_3_4_5, the AltOther field is:
# 1.2.3.4.5,<value of Other1a>
# Eg., in AltOther_1_2_3_4_6, the AltOther field is:
# 1.2.3.4.6,<value of Other2a><value of Other2b>
#
# Structure:
# 1) INSERT NAME - 'AltOther_' + <n1_n2_n3_n4_n5>
# 2) a hidden INPUT field with the same name as the INSERT NAME. It
#    is used to hold the AltOther field value to be included in the
#    certificate.
# 3) input field(s), with substitution variables [optfield] and [readonly].
#    [optfield] is used to control whether the field is an optional field.
#    [readonly] is used to control the display mode of the field: if the
#    field is on a web page requesting input, it will be assigned with
#    NULL, if it is on a web page for displaying request/certificate AltOther
#    information, it will be assigned with the HTML attribute 'READONLY'.
# 4) a hidden INPUT field with name 'altrawstring_' + <n1_n2_n3_n4_n5>.
#    Its value is a substitution variable [altrawvalue] which
#    is used to hold the result of the returned value of the AltOther,
#    excluding the OID and the comma. This is used for displaying the
#    AltOther information in a request or a certificate after it is
#    generated.
# 5) a JavaScript which will be called at load time. It contains the
#    parsing logic to parse the result obtained in 4) back into
#    individual input field(s) when the AltOther information
#    is displayed. Make sure the parsing logic matches the input
#    field(s) format.
# 6) a JavaScript function with the name built with a string
#    'ValidAltOther_' + <n1_n2_n3_n4_n5>. The name must be of this format.
#    Unlike the other validate functions in the other INSERTs which validate
#    user input(s) only, it also sets the variable specified in 2) above.
#    It concatenates the OID(n1.n2.n3.n4.n5) and value(s) of all the input
#    field(s).
#    You may customize different validation logic needed to validate
#    the input field(s).
#    The validation logic shown in the samples include:
#    - validate the required field(s) is/are filled
#    - validate the length of the input(s)
#    - pad the optional field(s) with preset value(s), if there is more
#      than one input field
# ======================================================================
#
# ======================================================================
# Sample AltOther INSERT with one input field
# ======================================================================
<INSERT NAME=AltOther_1_2_3_4_5>
#@LWC
<div aria-label="AltOther_1_2_3_4_5" role="region">
<INPUT NAME="AltOther_1_2_3_4_5" TYPE="hidden" maxlength="255">

<p> Other Name for alternate name: <BR>
<p> <LABEL for="other1afield">Customer's account number (11 digits)
[optfield] </LABEL> <BR>
<INPUT NAME="Other1a" TYPE="text" SIZE=11 maxlength="11" [readonly]
id="other1afield">

<INPUT NAME="altrawstring_1_2_3_4_5" TYPE="hidden" VALUE="[altrawvalue]">

<SCRIPT LANGUAGE="JavaScript">
<!--
//This is the script that will be called at load time.
var form=document.forms[0]
if (form.altrawstring_1_2_3_4_5.value.length > 0) {
  //The name 'Other<x>' needs to match with the above INPUT NAME.
  //Substr(start position, length)
  form.Other1a.value=form.altrawstring_1_2_3_4_5.value.substr(0,11)
}
//-->
</SCRIPT>

<SCRIPT LANGUAGE="JavaScript">
<!--
//This is the validation script
function ValidAltOther_1_2_3_4_5(frm){
 if (("[optfield]" == "" && frm.Other1a.value.length != 11) ||
     ("[optfield]" !="" && frm.Other1a.value != "" && frm.Other1a.value.length != 11)) {
```

```
   alert("Enter 11 digit account number.");
   frm.Other1a.focus();
   return false;
  }

  //Build the entire AltOther field.
  if (frm.Other1a.value != "")
   frm.AltOther_1_2_3_4_5.value = "1.2.3.4.5," + frm.Other1a.value ;
  else
   frm.AltOther_1_2_3_4_5.value = "";
  return true;
 }
 //-->
 </SCRIPT>
 </div>
 </INSERT>

 # ======================================================================
 # Sample AltOther INSERT with two input fields
 # ======================================================================
 <INSERT NAME=AltOther_1_2_3_4_6>
 #@LWC
 <div aria-label="AltOther_1_2_3_4_6" role="region">
 <INPUT NAME="AltOther_1_2_3_4_6" TYPE="hidden" maxlength="255">

 <p> Other Name for alternate name: <BR>
 <p> <LABEL for="other2afield">Customer's driver license number (9 digits)
 [optfield] </LABEL> <BR>
 <INPUT NAME="Other2a" TYPE="text" SIZE=9 maxlength="9" [readonly]
 id="other2afield">
 <p> <LABEL for="other2bfield">Customer's driver license expiration date
 (yyyymmdd) [optfield] </LABEL> <BR>
 <INPUT NAME="Other2b" TYPE="text" SIZE=8 maxlength="8" [readonly]
 id="other2bfield">

 <INPUT NAME="altrawstring_1_2_3_4_6" TYPE="hidden" VALUE="[altrawvalue]">

 <SCRIPT LANGUAGE="JavaScript">
 <!--
 //This is the script that will be called at load time.
 var form=document.forms[0]
 if (form.altrawstring_1_2_3_4_6.value.length > 0) {
   //The name 'Other<x>' needs to match with the above INPUT NAME.
   //Substr(start position, length)
   form.Other2a.value=form.altrawstring_1_2_3_4_6.value.substr(0,9)
   form.Other2b.value=form.altrawstring_1_2_3_4_6.value.substr(9,8)
 }
 //-->
 </SCRIPT>

 <SCRIPT LANGUAGE="JavaScript">
 <!--
 //This is the validation script
 function ValidAltOther_1_2_3_4_6(frm){
  if (("[optfield]" == "" && frm.Other2a.value.length != 9) ||
      ("[optfield]" !="" && frm.Other2a.value != "" && frm.Other2a.value.length != 9)) {
   alert("Enter 9 digit license number.");
   frm.Other2a.focus();
   return false;
  }
  if (("[optfield]" == "" && frm.Other2b.value.length != 8) ||
      ("[optfield]" !="" && frm.Other2b.value != "" && frm.Other2b.value.length != 8)) {
   alert("Enter date format yyyymmdd.");
   frm.Other2b.focus();
   return false;
  }
  if ("[optfield]" =="" && frm.Other2a.value == "" &&
       frm.Other2b.value == "") {
   alert("You must input at least one of the fields.");
   frm.Other2a.focus();
   return false;
  }

  //Pad the empty field with desired value for optional fields
  if ("[optfield]" !="") {
    if (frm.Other2a.value == "" && frm.Other2b.value != "") {
        frm.Other2a.value = "000000000";
    }
    else if (frm.Other2b.value == "" && frm.Other2a.value != "") {
        frm.Other2b.value = "00000000";
    }
  }
  //Build the entire AltOther field.
  if (frm.Other2a.value != "" && frm.Other2b.value != "")
   frm.AltOther_1_2_3_4_6.value = "1.2.3.4.6," + frm.Other2a.value +
                        frm.Other2b.value;
  else
   frm.AltOther_1_2_3_4_6.value = "";
  return true;
 }
 //-->
 </SCRIPT>
 </div>
 </INSERT>


 <INSERT NAME=UnstructName>
 <div role="region" aria-label="Unstructured Name">
 <p> <LABEL for="unstructnamefield">Unstructured device name
 [optfield] </LABEL> <BR>
 <INPUT NAME="UnstructName" TYPE="text" SIZE=64 maxlength="64"
 id="unstructnamefield">
 <SCRIPT LANGUAGE="JavaScript">
 <!--
 function ValidUnstructName(frm){
 if ("[optfield]" == "" && frm.UnstructName.value == "") {
 alert("Enter required field."); frm.UnstructName.focus();
 return false;
 }
 return true;
 }
 //-->
 </SCRIPT>
 </div>
 </INSERT>

 <INSERT NAME=UnstructAddr>
 <div role="region" aria-label="Unstructured Address">
 <p> <LABEL for="unstructaddrfield">Unstructured device address
 [optfield] </LABEL> <BR>
 <INPUT NAME="UnstructAddr" TYPE="text" SIZE=64 maxlength="64"
 id="unstructaddrfield">
 <SCRIPT LANGUAGE="JavaScript">
 <!--
 function ValidUnstructAddr(frm){
 if ("[optfield]" == "" && frm.UnstructAddr.value == "") {
 alert("Enter required field."); frm.UnstructAddr.focus();
 return false;
 }
 return true;
 }
 //-->
 </SCRIPT>
 </div>
 </INSERT>


 <INSERT NAME=SerialNumber>
 <div role="region" aria-label="Serial Number">
 <p> <LABEL for="serialnumberfield">Device serial number
 [optfield] </LABEL> <BR>
 <INPUT NAME="SerialNumber" TYPE="text" SIZE=64 maxlength="64"
 id="serialnumberfield">
 <SCRIPT LANGUAGE="JavaScript">
 <!--
 function ValidSerialNumber(frm){
 if ("[optfield]" == "" && frm.SerialNumber.value == "") {
 alert("Enter required field."); frm.SerialNumber.focus();
 return false;
 }
 return true;
 }
 //-->
 </SCRIPT>
 </div>
 </INSERT>

 # ==================================================================
 # Sample AltOther INSERT for Windows Logon
 # ==================================================================
 <INSERT NAME=AltOther_1_3_6_1_4_1_311_20_2_3>
 #@LWC
 <div aria-label="AltOther_1_3_6_1_4_1_311_20_2_3" role="region">
```

```
<INPUT NAME="AltOther_1_3_6_1_4_1_311_20_2_3" TYPE="hidden" maxlength="255">

<p> Other Name for alternate name: <BR>
<p> <LABEL for="other3afield">User Principal Name (max 50 chars)
[optfield] </LABEL> <BR>
<INPUT NAME="Other3a" TYPE="text" SIZE=50 maxlength="50" [readonly]
id="other3afield">

<INPUT NAME="altrawstring_1_3_6_1_4_1_311_20_2_3" TYPE="hidden"
       VALUE="[altrawvalue]">

<SCRIPT LANGUAGE="JavaScript">
<!--
//This is the script that will be called at load time.
var form=document.forms[0]
if (form.altrawstring_1_3_6_1_4_1_311_20_2_3.value.length > 0) {
  //The name 'Other<x>' needs to match with the above INPUT NAME.
  //Substr(start position, length)
  form.Other3a.value=form.altrawstring_1_3_6_1_4_1_311_20_2_3.value
}
//-->
</SCRIPT>

<SCRIPT LANGUAGE="JavaScript">
<!--
//This is the validation script
function ValidAltOther_1_3_6_1_4_1_311_20_2_3(frm){
 if ("[optfield]" == "" && frm.Other3a.value == "") {
  alert("Enter User Principal Name.");
  frm.Other3a.focus();
  return false;
 }

 if (frm.Other3a.value != "") {
   // Verify the UPN has an atsign('@'), and that it is not
   // in either the first or last character position.
   if ((frm.Other3a.value.indexOf("@") <= 0)
    || (frm.Other3a.value.indexOf("@") == frm.Other3a.value.length-1)
      ) {
     alert("Enter User Principal Name in the form of id@domain.");
     frm.Other3a.focus();
     return false;
   }
 }

 //Build the entire AltOther field.
 if (frm.Other3a.value != "")
  frm.AltOther_1_3_6_1_4_1_311_20_2_3.value = "1.3.6.1.4.1.311.20.2.3,"
                                            + frm.Other3a.value;
 else
  frm.AltOther_1_3_6_1_4_1_311_20_2_3.value = "";
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

# ======================================================================
# Sample Custom Extension INSERT  @LHA
# ======================================================================
# Here it shows one 'CustomExt' INSERT. You may add as many as you need.
# Structure:
#
# 1) INSERT NAME with format:
#      the string 'CustomExt_'||_ separated OID||'_'||
#      the critical flag in upper case: 'N' or 'C'||'_'||
#      the encode type in upper case: 'INT','PRT','IA5','BMP' or 'OCT'
# 2) a hidden INPUT field with the same name as the INSERT NAME. It
#    is used to hold the CustomExt field value to be included in the
#    certificate.
# 3) input field(s), with substitution variables [optfield] and [readonly].
#    [optfield] is used to control whether the field is an optional field.
#    [readonly] is used to control the display mode of the field: if the
#    field is on a web page requesting input, it will be assigned with
#    NULL, if it is on a web page for displaying request/certificate CustomExt
#    information, it will be assigned with the HTML attribute 'READONLY'.
# 4) a hidden INPUT field with name similar to the INSERT name, except
#    'CustomExt_' is replaced by 'customstring_'
#    Its value is a substitution variable [custvalue] which
#    is used to hold the result of the returned value of the CustomExt,
#    excluding the OID, the critical flag, the encode type and the
#    commas. This is used for displaying the CustomExt information in
#    a request or a certificate after it is generated.
# 5) a JavaScript which will be called at load time. It contains the
#    parsing logic to parse the result obtained in 4) back into
#    individual input field(s) when the CustomExt information
#    is displayed. Make sure the parsing logic matches the input
#    field(s) format.
# 6) a validation JavaScript function with the name built with a string
#    similar to the INSERT name except the part 'CustomExt' is
#    replaced by 'ValidCustomExt_'
#    Unlike the other validate functions in the other INSERTs which validate
#    user input(s) only, it also sets the variable specified in 2) above.
#    It concatenates the OID, the critical flag, the encode type,
#    and the value(s) of all the input field(s).
#    You may customize different validation logic needed to validate
#    the input field(s).
#    The validation logic shown in the sample include:
#    - validate the required field(s) is/are filled
#    - validate '@' is not in the input
# ===================================================================
<INSERT NAME=CustomExt_1_3_6_1_4_1_311_20_2_N_BMP>
#@LWC
<div aria-label="CustomExt_1_3_6_1_4_1_311_20_2_N_BMP" role="region">
<INPUT NAME="CustomExt_1_3_6_1_4_1_311_20_2_N_BMP" TYPE="hidden" maxlength="16">

<p> Custom Extension: <BR>
<p> <LABEL for="custom1field">Certificate template name
[optfield] </LABEL> <BR>
<INPUT NAME="Custom1" TYPE="text" SIZE=16 maxlength="16" [readonly]
id="custom1field">

<INPUT NAME="customstring_1_3_6_1_4_1_311_20_2_N_BMP" TYPE="hidden" VALUE="[custvalue]">

<SCRIPT LANGUAGE="JavaScript">
<!--
//This is the script that will be called at load time.
var form=document.forms[0]
if (form.customstring_1_3_6_1_4_1_311_20_2_N_BMP.value.length > 0) {
  //The name 'Custom<x>' needs to match with the above INPUT NAME.
  //Substr(start position, length)
  form.Custom1.value=form.customstring_1_3_6_1_4_1_311_20_2_N_BMP.value.substr(0,16)
}
//-->
</SCRIPT>

<SCRIPT LANGUAGE="JavaScript">
<!--
//This is the validation script
function ValidCustomExt_1_3_6_1_4_1_311_20_2_N_BMP(frm){
 if ("[optfield]" == "" && frm.Custom1.value == "") {
  alert("Enter Certificate Template Name.");
  frm.Custom1.focus();
  return false;
 }
 if (frm.Custom1.value != "") {
   // Verify the input value, eg. it must not contain an atsign('@')
   if ((frm.Custom1.value.indexOf("@") >= 0))
    {
      alert("Invalid format - no '@' allowed.");
      frm.Custom1.focus();
      return false;
    }
 }

 //Build the Custom Extension field.
 if (frm.Custom1.value != "")
  frm.CustomExt_1_3_6_1_4_1_311_20_2_N_BMP.value =
      "1.3.6.1.4.1.311.20.2,N,BMP," + frm.Custom1.value;
 else
  frm.CustomExt_1_3_6_1_4_1_311_20_2_N_BMP.value = "";
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>


# ===================================================================
#
# non-X.509 certificate request fields (INSERTs)
#
```

```
# ======================================================================
#
<INSERT NAME=UserId>
<div role="region" aria-label="User ID">
<p> <LABEL for="safuseridfield">Owning SAF User ID [optfield]
</LABEL> <BR>
<INPUT NAME="UserId" TYPE="text" SIZE=8 maxlength="8"
id="safuseridfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidUserId(frm){
 if ("[optfield]" == "" && frm.UserId.value == "") {
  alert("Enter required field."); frm.UserId.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=Label>
<div role="region" aria-label="Label">
<p> <LABEL for="labelfield">Label assigned to certificate being
requested [optfield] </LABEL> <BR>
<INPUT NAME="Label" TYPE="text" SIZE=32 maxlength="32"
id="labelfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidLabel(frm){
 if ("[optfield]" == "" && frm.Label.value == "") {
  alert("Enter required field."); frm.Label.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=Requestor>
<div role="region" aria-label="Requestor">
<p> <LABEL for="requestorfield">Your name for tracking this request
[optfield] </LABEL> <BR>
<INPUT NAME="Requestor" TYPE="text" SIZE=32 maxlength="32"
id="requestorfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidRequestor(frm){
 if ("[optfield]" == "" && frm.Requestor.value == "") {
  alert("Enter required field."); frm.Requestor.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=Requestor2>
<div role="region" aria-label="Requestor Email">
<p> <LABEL for="requestor2field">Enter the requestor's email address
[optfield] </LABEL> <BR>
<INPUT NAME="Requestor" TYPE="text" SIZE=32 maxlength="32"
id="requestor2field">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidRequestor2(frm){
 if ("[optfield]" == "" && frm.Requestor.value == "") {
  alert("Enter required field."); frm.Requestor.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>
```

```
<INSERT NAME=PassPhrase>  7
<div role="region" aria-label="Password">
<p> <LABEL for="passphrasefield">Pass phrase for securing this request.
You will need to supply this value when retrieving your certificate
[optfield] </LABEL> <BR>
#@DKC
<INPUT NAME="PassPhrase" TYPE="password" SIZE=32 maxlength="32"
id="passphrasefield" autocomplete="off"> <BR>
<p> <LABEL for="passphrase2field">Reenter your pass phrase to
confirm </LABEL> <BR>
#@DKC
<INPUT NAME="ConfirmPassPhrase" TYPE="password" SIZE=32
 maxlength="32" id="passphrase2field" autocomplete="off">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidPassPhrase(frm){
 if ("[optfield]" == "" && frm.PassPhrase.value == "") {
  alert("Enter required field."); frm.PassPhrase.focus();
  return false;
 }
 if ("[optfield]" == "" && frm.ConfirmPassPhrase.value == "") {
  alert("Reenter the pass phrase."); frm.ConfirmPassPhrase.focus();
  return false;
 }
 if (frm.PassPhrase.value != frm.ConfirmPassPhrase.value) {
  alert("Passwords don't match. Reenter."); frm.PassPhrase.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=ChallengePassPhrase>
<div role="region" aria-label="Pass Phrase">
<p> <LABEL for="challengefield">If you specified a pass phrase when
submitting the certificate request, type it here, exactly as you
typed it on the request form </LABEL> <BR>
#@DKC
<INPUT NAME="ChallengePassPhrase" TYPE="password" SIZE=32
maxlength="32" id="challengefield" autocomplete="off">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidChallengePassPhrase(frm){
 if ("[optfield]" == "" && frm.ChallengePassPhrase.value == "") {
  alert("Enter required field."); frm.ChallengePassPhrase.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

#@DKC
<INSERT NAME=-ChallengePassPhrase2>
<div role="region" aria-label="Re-enter Pass Phrase">
<p> <LABEL for="challenge2field">Enter the same pass phrase as on
the request form  </LABEL> <BR>
<INPUT NAME="ChallengePassPhrase" TYPE="password" SIZE=32
maxlength="32" id="challenge2field" autocomplete="off">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidChallengePassPhrase2(frm){
 if ("[optfield]" == "" && frm.ChallengePassPhrase.value == "") {
  alert("Enter required field."); frm.ChallengePassPhrase.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

# HostIdMap is a repeatable field. If more than one is needed, a
# separate INSERT, which can be modelled from this one, is needed.
# See INSERT NAME=HostIdMap2 for an example.
<INSERT NAME=HostIdMap>
```

```
<div role="region" aria-label="Host ID Map">
<p> <LABEL for="hostidmapfield">HostIdMapping Extension value in
subject-id@host-name form [optfield] </LABEL> <BR>
<INPUT NAME="HostIdMap" TYPE="text" SIZE=100 maxlength="100"
id="hostidmapfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidHostIdMap(frm){
 if ("[optfield]" == "" && frm.HostIdMap.value == "") {
  alert("Enter required field."); frm.HostIdMap.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=HostIdMap2>
<div role="region" aria-label="Host ID Map 2">
<p> <LABEL for="hostidmap2field">HostIdMapping Extension value in
subject-id@host-name form [optfield] </LABEL> <BR>
<INPUT NAME="HostIdMap2" TYPE="text" SIZE=100 maxlength="100"
id="hostidmap2field">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidHostIdMap2(frm){
 if ("[optfield]" == "" && frm.HostIdMap2.value == "") {
  alert("Enter required field."); frm.HostIdMap2.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=-TransactionId>
<div role="region" aria-label="Transaction ID">
<p> <LABEL for ="TransactionIdfield">Enter the assigned transaction
ID [optfield]</LABEL> <BR>
<INPUT NAME="TransactionId" TYPE="text" SIZE=56 maxlength="56"
VALUE="[transactionid]" id = "TransactionIdfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidTransactionId(frm){
 if ("[optfield]" == "" && frm.TransactionId.value == "") {
  alert("Enter required field."); frm.TransactionId.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=NotifyEmail>
<div role="region" aria-label="Notify Email">
<p> <LABEL for="notifyemailfield">Email address for notification
purposes [optfield]  </LABEL> <BR>
<INPUT NAME="NotifyEmail" TYPE="text" SIZE=64 MAXLENGTH="64"
id="notifyemailfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidNotifyEmail(frm){
 if ("[optfield]" == "" && frm.NotifyEmail.value == "") {
  alert("Enter required field."); frm.NotifyEmail.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

#@LEA
<INSERT NAME=-RecoverEmail>
<div role="region" aria-label="Recover Email">
```

```
<p> <LABEL for="recoveremailfield">Enter the email address when the
original certificate was requested [optfield] </LABEL> <BR>
<!-- @DIC -->
<INPUT NAME="RecoverEmail" TYPE="text" MAXLENGTH=32 SIZE=32
id="recoveremailfield" VALUE="[requestoremail]">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidRecoverEmail(frm){
 if ("[optfield]" == "" && frm.RecoverEmail.value == "") {
  alert("Enter required field."); frm.RecoverEmail.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

#@LGA
<INSERT NAME=-RecoverEmail2>
<div role="region" aria-label="Recover Email Two">
<p> <LABEL for="recoveremail2field">Enter the email address when the
original certificate was requested [optfield] </LABEL> <BR>
<INPUT NAME="RecoverEmail" TYPE="text" MAXLENGTH=32 SIZE=32
VALUE="[requestor]" id="recoveremail2field">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidRecoverEmail(frm){
 if ("[optfield]" == "" && frm.RecoverEmail.value == "") {
  alert("Enter required field."); frm.RecoverEmail.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=SelectCADomain>
<div role="region" aria-label="Select CA Domain">
<p> <LABEL for="selectcadomfield">Select the CA domain to work with
</LABEL>
<SELECT NAME="domain" id="selectcadomfield">
# rename and replicate the following line for every CA domain and
# determine which one should be SELECTED by default, if any
<OPTION VALUE="Customers" SELECTED>Customers
</SELECT>
</div>
</INSERT>

# Changed name of insert from SmartCardNS to PublicKey2NS to match
# CGI scripts                                                      @LDC
# Added the confirmaton dialog box and <keygen> html tag           @LJA
<INSERT NAME=PublicKey2NS>
<div role="region" aria-label="Public Key2 NS">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidSmartcard(frm){
   var message = "Make sure you have Smart Card(s) installed and loaded \
on the browser. Choose the appropriate Smart Card device from the upcoming \
option list. If you cannot confirm a smart card is configured, click Cancel."

 var response = confirm(message);
 if(response == true)
    return true;
 else
 {
   history.back();
   return false;
 }
}
//-->
</SCRIPT>
<p><LABEL for="keygentag2"> Select a key size</LABEL>
<KEYGEN NAME="PublicKey" id="keygentag2">
</div>
</INSERT>

# Changed name of insert from SmartCardIE to PublicKey2IE to match
# CGI scripts                                                      @LDC
```

```
<INSERT NAME=PublicKey2IE>
<div role="region" aria-label="Public Key2 IE">
#Converted VBScript to JavaScript                              @LUC
<SCRIPT LANGUAGE="JavaScript">
<!--
function SendReq(){
  var pkcs10data,DN,i,Message,CommonName;
  var objEnroll;
  DN= "";
  CommonName= "Unspecified Distinguished Name";
  DN= "CN=" + CommonName + ";";
  pkcs10data = "";
  try{
    objEnroll = g_objWCF.CreateObject("X509Enrollment.CX509Enrollment");
  }catch(err){
    objEnroll = null;
  }
  if(objEnroll !== null && typeof objEnroll === 'object'){
    // CertEnroll API
    var objPrivateKey;
    var objRequest;
    var objName;
    var objHash;                                                 // @LWA
    try{
      objPrivateKey = g_objWCF.CreateObject("X509Enrollment.CX509PrivateKey");
    }catch(err){
      Message = "Error creating Private Key object: \n" + err.description;
      alert(Message);
      return;
    }
    try{
      objRequest = g_objWCF.CreateObject("X509Enrollment.CX509CertificateRequestPkcs10");
    }catch(err){
      Message = "Error creating Request object: \n" + err.description;
      alert(Message);
      return;
    }
    i = document.getElementById("smartcardcspfield").options.selectedIndex;
    objPrivateKey.ProviderName = document.getElementById("smartcardcspfield").options(i).text;
    objPrivateKey.ProviderType = document.getElementById("smartcardcspfield").options(i).value;
    objPrivateKey.KeySpec      = 1; // XCN_AT_KEYEXCHANGE
    objPrivateKey.Length       = 2048; // KeySize = 2048            @LWA
    try{
      objRequest.InitializeFromPrivateKey( 1, objPrivateKey, "");
    }catch(err){
      Message = "Error initializing request from private key " + err.description;
      alert(Message);
      return;
    }
    try{
      objName = g_objWCF.CreateObject("X509Enrollment.CX500DistinguishedName");
    }catch(err){
      Message = "Error creating X500DistinguishedName object: \n" + err.description;
      alert(Message);
      return;
    }
    try{
      objName.Encode(DN);
    }catch(err){
      Message = "Error encoding the subject distinguished name \n" + err.description;
      alert(Message);
      return;
    }
    try{
      objRequest.Subject = objName;
    }catch(err){
      Message = "Error setting the subject name in request " + err.description;
      alert(Message);
      return;
    }
    //setting signature algorithm SHA256 in request                 @LWA
    try {                                                        // @LWA
      objHash = g_objWCF.CreateObject("X509Enrollment.CObjectId");
      objHash.InitializeFromAlgorithmName( 4           //XCN_CRYPT_SIGN_ALG_OID_GROUP_ID
                                        , 0x00000000  //XCN_CRYPT_OID_INFO_PUBKEY_ANY
                                        , 0x00000000  //AlgorithmFlagsNone
                                        ,"SHA256");             // @LWA
      objRequest.HashAlgorithm = objHash;                        // @LWA
    } catch(err) {
      Message = "Error setting the signature algorithm in request "
                + err.description;                               // @LWA
      alert(Message);                                            // @LWA
      return;                                                    // @LWA
    }                                                            // @LWA


    try{
      objEnroll.InitializeFromRequest( objRequest )
    }catch(err){
      Message = "Error initializing Enrollment object from request: " + err.description;
      alert(Message);
      return;
    }
    pkcs10data = objEnroll.CreateRequest(1); // XCN_CRYPT_STRING_BASE64
    //11@LVD
  }
  document.CertReq.PublicKey.value = pkcs10data;
  if(pkcs10data.length <= 0){
    alert("PKCS10 Creation Failed");
  }
}
//-->
</SCRIPT>

<p> <LABEL for="smartcardcspfield">Select from the following installed
smartcard providers </LABEL> <br>
<select name="CSP" id="smartcardcspfield">
#Converted VBScript to JavaScript                                   @LUC
<SCRIPT LANGUAGE="JavaScript">
<!--
function LoadCSPs(){
  try {
    var i;
    var csp;
    var sv;
    //Modifications for CertEnroll API enrollment process.
    var objCSPs;
    var oOption;
    var provider;
    var errmsg;
    try{
      objCSPs = g_objWCF.CreateObject("X509Enrollment.CCspInformations");
    }catch(err){
      objCSPs = null;
      alert("Failed to load CSPs"); //@LVA
      return;                       //@LVA
    }
    if(objCSPs !== null && typeof objCSPs === 'object'){
      //CertEnroll APIs
      objCSPs.AddAvailableCsps();
      for(i=0 ; i < objCSPs.Count;i++){
        //Only include Legacy(Crypto API) providers at this time
        if(objCSPs.ItemByIndex(i).LegacyCsp){
          provider = objCSPs.ItemByIndex(i).Name.toLowerCase();
          if((provider.indexOf("smart") > 0) || (provider.indexOf("card") > 0)){
            oOption = document.createElement("OPTION");
            oOption.text = objCSPs.ItemByIndex(i).Name;
            oOption.value = objCSPs.ItemByIndex(i).Type;
            document.getElementById("smartcardcspfield").add(oOption);
          }
        }
      }
      //36@LVD
    }
  }catch (e) {
    alert("Failed to load CSPs");
  }
}
//-->
</SCRIPT>
</select>

<input type="hidden" name="PublicKey" value="">
<p>
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidSmartcard(frm){
 SendReq()
 if (document.CertReq.PublicKey.value == "")
  return false;
 else
  return true;
}
//-->
```

```
</SCRIPT>
</div>
</INSERT>

#####################################################################
#
# This INSERT is for preregistration only
#
#####################################################################
<INSERT NAME=ClientName>
<div role="region" aria-label="Client Name">
<p> <LABEL for="clientnamefield">The name of the person or device that
the certificate represents  </LABEL> <BR>
<INPUT NAME="ClientName" TYPE="text" SIZE=64 maxlength="64"
id="clientnamefield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidClientName(frm){
if (frm.ClientName.value == "") {
alert("Enter required field."); frm.ClientName.focus();
return false;
}
return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

#####################################################
#                                                   #
# This INSERT is for PKI generated key request only #
# @LIC                                              #
#####################################################
<INSERT NAME=KeySize>
 <div role="region" aria-label="Key Size Field">
<p> <LABEL for="keysizefield">Select the key type and key size
</LABEL> <BR>
<SELECT NAME="KeySize" id="keysizefield">
#1@LQD
<OPTION VALUE="RSA - 1024">RSA - 1024
<OPTION VALUE="RSA - 2048">RSA - 2048
<OPTION VALUE="RSA - 4096">RSA - 4096
<OPTION VALUE="NISTECC - 192">NISTECC - 192
<OPTION VALUE="NISTECC - 224">NISTECC - 224
<OPTION VALUE="NISTECC - 256">NISTECC - 256
<OPTION VALUE="NISTECC - 384">NISTECC - 384
<OPTION VALUE="NISTECC - 521">NISTECC - 521
<OPTION VALUE="BPECC - 160">BPECC - 160
<OPTION VALUE="BPECC - 192">BPECC - 192
<OPTION VALUE="BPECC - 224">BPECC - 224
<OPTION VALUE="BPECC - 256">BPECC - 256
<OPTION VALUE="BPECC - 320">BPECC - 320
<OPTION VALUE="BPECC - 384">BPECC - 384
<OPTION VALUE="BPECC - 512">BPECC - 512
</SELECT>
<SCRIPT LANGUAGE="JavaScript">
<!--
// Because keysize is a single select field, it will always have a
// value, therefore we do not need to check if a required field was
// provided.  Also, we assume that only valid values are included in
// the selection options.

function ValidKeySize(frm)
{
return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

#####################################################
#                                                   #
# These INSERTs are used to assist the recovery of  #
# the certificate whose key is generated by PKI.    #
# You can add as many as you want. Start from       #
# Security1, then Security2 ... Securityn and so on.#
# These are meant to be used by the GENCERT/REQCERT #
# and QRECOVER exits.                               #
# The number of these questions must match to that  #
# handled by the exits.                             #
# @LEA                                              #
#####################################################
<INSERT NAME=Security1>
<div role="region" aria-label="Security Question One">
<p> <LABEL for="security1field">What's the intended use of this
certificate? [optfield]  </LABEL> <BR>
<INPUT NAME="Security1" TYPE="text" SIZE=100 maxlength="100"
id="security1field">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidSecurity1(frm){
 if ("[optfield]" == "" && frm.Security1.value == "") {
  alert("Enter required field."); frm.Security1.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

<INSERT NAME=Security2>
<div role="region" aria-label="Security Question Two">
<p> <LABEL for="security2field">What's the name of your elementary
school? [optfield] </LABEL> <BR>
<INPUT NAME="Security2" TYPE="text" SIZE=100 maxlength="100"
id="security2field">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidSecurity2(frm){
 if ("[optfield]" == "" && frm.Security2.value == "") {
  alert("Enter required field."); frm.Security2.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>

####################################################################
#                                                                  #
#                   Additional section                             #
#                                                                  #
####################################################################

<INSERT NAME=-copyright>
<META HTTP-EQUIV="Content-Type" content="text/html; charset=ISO-8859-1">
<!--
/******************************************************************/
/*                                                                */
/* Licensed Materials - Property of IBM                           */
/* 5650-ZOS                                                       */
/* Copyright IBM Corp. 2001, 2019                                 */
/*                                                                */
/******************************************************************/
-->
</INSERT>
<INSERT NAME=-pagefooter>
<div role="region" aria-label="Contact Email">
<A HREF="mailto:webmaster@your-company">
email: webmaster@your-company.com</A>
</div>
</INSERT>

#####################################################################
#
# This INSERT is for Installation of Auto Renewed certificate     // @DFA
#
#####################################################################
#Converted VBScript to JavaScript                                74@LUC
<INSERT NAME=InstallCert>
<SCRIPT LANGUAGE="JavaScript">
<!--
function installCert(){
  //------------------------------------------------------------------------------
  //Function Call to install Certificate after the creation of the renewal request
  //------------------------------------------------------------------------------
  var msg;
  var pkcs7data, errmsg, rc;
  // Added for CertEnroll API processing.
  var objEnroll;
  var temp;
  var beginlen;
  var beginpos;
  var begintag;
  try{
    var pkcs7data = document.getElementById("b64cert").value;
    //Remove begin certificate tag
    begintag = "-----BEGIN CERTIFICATE-----"
    if(pkcs7data.indexOf(begintag) >= 0){
      temp = pkcs7data;
      beginlen = begintag.length;
      beginpos = temp.indexOf(begintag) + beginlen;
      pkcs7data = temp.substring(beginpos);
    }
    // CertEnroll.dll API additions follow.
    try{
      objEnroll = g_objWCF.CreateObject("X509Enrollment.CX509Enrollment");
    }catch(err){
      objEnroll = null;
    }
    if(objEnroll !== null && typeof objEnroll === 'object'){
      try{
        //CertEnroll APIs
        objEnroll.Initialize(1);  // ContextUser
      }catch(err){
        errmsg = "Error Initializing Enrollment object. " + err.description;
        alert(errmsg);
        return 1;
      }
      try{
        objEnroll.InstallResponse(0, pkcs7data, 1, "");
      }catch(err){
        errmsg = "Error Installing Response. " + err.description;
        alert(errmsg);
        return 1;
      }
      //10@LVD
    }
  }catch(err){
    errmsg = "Your new certificate failed to install. " +
     "Please ensure that you are using the same browser " +
     "that you used when making the certificate request. " +
     "Also ensure that PKI ActiveX is installed.";
    alert(errmsg);
    return;
  }
  errmsg = "Your new certificate installed successfully.";
  alert(errmsg);
  return 0;
}
// -->
</SCRIPT>

<SCRIPT LANGUAGE="JavaScript">
<!--
// Function to create the renewal request. If successful, go ahead and
// call the VB script function to install the certificate on the browser
function InstallCertificate()
{
  // return failure if certificate not present
  if(document.getElementById("b64cert").value == "")
  {
    alert("Auto Renew Certificate Install failed - Missing required base64 certificate");
    document.getElementById("b64cert").focus();
    return false;
  }

  //Call to create renewal request
  var result = ValidRenewKeySet();
  if(result == false)
  {
//7@LVD
         return false;
//@LVD
  }
  else
  {
        var res1 = installCert();
        return res1;
  }
```

```
}
//-->
</SCRIPT>
</INSERT>
######################################################################
#
# This INSERT is BusinessCat                                        @LOA
#
######################################################################
<INSERT NAME=BusinessCat>
<div role="region" aria-label="Business Category">
<p> <LABEL for="businesscatfield">Business Category [optfield] </LABEL>
<BR>
<INPUT NAME="BusinessCat" TYPE="text" SIZE=64  maxlength="64"
id="businesscatfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidBusinessCat(frm){
 if ("[optfield]" == "" && frm.BusinessCat.value == "") {
  alert("Enter required field."); frm.BusinessCat.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>
######################################################################
#
# This INSERT is JurLocality                                        @LOA
#
######################################################################
<INSERT NAME=JurLocality>
<div role="region" aria-label="Jurisdiction of Incorporation Locality Name">
<p> <LABEL for="jurlocalityfield">Jurisdiction of Incorporation Locality
Name [optfield] </LABEL> <BR>
<INPUT NAME="JurLocality" TYPE="text" SIZE=64  maxlength="64"
id="jurlocalityfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidJurLocality(frm){
 if ("[optfield]" == "" && frm.JurLocality.value == "") {
  alert("Enter required field."); frm.JurLocality.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>
######################################################################
#
# This INSERT is JurStateProv                                       @LOA
#
######################################################################
<INSERT NAME=JurStateProv>
<div role="region" aria-label="Jurisdiction of Incorporation State or Province Name">
<p> <LABEL for="jurstateprovfield">Jurisdiction of Incorporation State or
Province Name [optfield] </LABEL> <BR>
<INPUT NAME="JurStateProv" TYPE="text" SIZE=64  maxlength="64"
id="jurstateprovfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidJurStateProv(frm){
 if ("[optfield]" == "" && frm.JurStateProv.value == "") {
  alert("Enter required field."); frm.JurStateProv.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>
######################################################################
#
# This INSERT is JurCountry                                         @LOA
#
######################################################################
<INSERT NAME=JurCountry>
```

```
<div role="region" aria-label="Jurisdiction of Incorporation Country Name">
<p> <LABEL for="jurcountryfield">Jurisdiction of Incorporation Country
Name [optfield] </LABEL> <BR>
<INPUT NAME="JurCountry" TYPE="text" SIZE=2  maxlength="2"
id="jurcountryfield">
<SCRIPT LANGUAGE="JavaScript">
<!--
function ValidJurCountry(frm){
 if ("[optfield]" == "" && frm.JurCountry.value == "") {
  alert("Enter required field."); frm.JurCountry.focus();
  return false;
 }
 return true;
}
//-->
</SCRIPT>
</div>
</INSERT>
#//6@LVD  XP header deleted
#####################################################################
#
# This INSERT is for loading Microsoft Enrollment Object for
# Windows Operating System                                       @DLA
#
#####################################################################
#//@LVC
<INSERT NAME=-ObjectHeaderIE>
#-- Create g_objWCF object for CertEnroll process
<OBJECT
  classid="clsid:884e2049-217d-11da-b2a4-000e7bbb2b09"
  id="g_objWCF">cert enroll object is not available
</OBJECT>
</INSERT>
#####################################################################
#
# This INSERT is for checking if the OS is WinXP, XP is no longer
# supported and will cause an error message to be displayied
# The check is only required for IE as FF will still work        @LVA
#
#####################################################################
</INSERT>
<INSERT NAME=-CheckXP>
<SCRIPT LANGUAGE="JavaScript">
<!--
function CheckXP(){
  var OS = navigator.userAgent;
  if ((OS.indexOf("Windows NT 5")!=-1)||(OS.indexOf("Windows NT 6.0")!=-1)){ //XP and Vista are
unsupported
    document.write("<HTML lang='en'><HEAD><TITLE>Page No Longer Supported</TITLE></HEAD>");
    document.write("<BODY><div role='main'><H1>This page is no longer supported for the current
Operating System.");
    document.write("Please access using a modern Operating System.<br>");
    document.write("</H1></div></BODY></HTML>");
  }
}
//-->
</SCRIPT>
</INSERT>
```

The numbers in the following list refer to the highlighted tags in the preceding excerpt of the INSERT section.

1. The `-requestok` INSERT has the logic to generate the certificate. If the certificate is successfully generated, a web page (whose main heading is "Request submitted successfully") is displayed. This web page includes the transaction ID.
2. The `-requestok` INSERT includes an ACTION that calls `caretrieve.rexx`, which allows the user to retrieve the certificate.
3. Alternately, if the request is not successful, the `-requestbad` INSERT gains control.
4. (The `caretrieve.rexx` CGI displays the RETRIEVECONTENT subsection (see list item "15" on page 184) HTML, which displays a web page that prompts the user for the transaction ID associated with the certificate request. The user enters the transaction ID (and any password) and clicks the **Continue** button, which calls `cagetcert.rexx`.) The `cagetcert.rexx` CGI calls `R_PKIServ` for EXPORT of the certificate. If the export is successful, `cagetcert.rexx` displays the HTML under the RETURNCERT subsection. (See list item "18" on page 185.)

5. The base64-encoded certificate is displayed on the web page by using the `[base64cert]` substitution variable.
6. This is a browser-qualified PublicKey INSERT for Internet Explorer.
7. Additional INSERTs are certificate field name INSERTs. These describe the fields using the HTML dialogs that are displayed on the web pages if the user is allowed to input these fields. For example, PassPhrase is a text field with a maximum length of 32 characters. The two-year PKI browser certificate for authenticating to z/OS allows the user to fill in this field. (`%%PassPhrase%%` is listed in the input fields; see list item "8" on page 184.)

Relationship between CGIs and the pkiserv.tmpl file

CGIs for the end-user web pages are execs that gain control when the end user clicks an action button - for example, the **Request certificate** button on the PKI Services home page. The CGIs read the `pkiserv.tmpl` file to determine the action to perform. They resolve substitution variables in the `pkiserv.tmpl` file.

The CGIs for the end-user web pages (including their directories) are:

- `/usr/lpp/pkiserv/PKIServ/ssl-cgi/camain.rexx`
- `/usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/catmpl.rexx`
- `/usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/auth/careq.rexx`
- `/usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/carecover.rexx`
- `/usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/caretrieve.rexx`
- `/usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/auth/cagetcert.rexx`
- `/usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/auth/cagetcert2.rexx`
- `/usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/cagorcvr.rexx`
- `/usr/lpp/pkiserv/PKIServ/clientauth-cgi-bin/cadisplay.rexx`
- `/usr/lpp/pkiserv/PKIServ/clientauth-cgi-bin/camodify.rexx`
- `/usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/installcert.rexx`

Note: `installcert.rexx` is invoked from a link in the certificate renewal email notification, not from a web page.

The following table summarizes the actions the CGIs perform:

Table 41. CGI actions for end-user web pages

CGI exec	Action	Sample web page
`camain.rexx`	• When the user clicks **Request certificate**, this CGI calls `catmpl.rexx`, passing it a parameter identifying the selected template. • The user can click **Pick up certificate** to go directly to `caretrieve.rexx` (if the certificate is already requested). • The user can click **Renew or revoke certificate** to go to `cadisplay.rexx`.	See "#unique_66/unique_66_Connect_42_mainpage" on page 376.
`catmpl.rexx`	• Displays web page coded in the HTML under the CONTENT subsection (of a TEMPLATE section). • When the user clicks **Submit certificate request**, this CGI passes template and field name parameters to `careq.rexx`. • When the user clicks **Retrieve your certificate**, this CGI passes control to `caretrieve.rexx`.	See Figure 44 on page 386.

Table 41. CGI actions for end-user web pages (continued)

CGI exec	Action	Sample web page
`carecover.rexx`	• Displays web page coded in the HTML under the RECOVERCONTENT subsection (of a TEMPLATE section). • This HTML prompts the user to enter the requestor name and passphrase if the user entered one when requesting the certificate. • When the user clicks **Recover Certificate**, this CGI passes the requestor name and passphrase to `cagryrcvr.rexx`.	See Figure 56 on page 400
`careq.rexx`	• Processes field names under the APPL subsection (of a TEMPLATE section). **Note:** Depending on the template, the field names can be: – `UserId` only – `UserId` and `HostIdMap`. • Processes hardcoded field names under the CONSTANT subsection (of a TEMPLATE section). • Depending on the results, displays web page coded in the HTML under the SUCCESSCONTENT or FAILURECONTENT subsection (of a TEMPLATE section): – The SUCCESSCONTENT subsection includes a **Continue** button the user can click to continue to `caretrieve.rexx`.	See Figure 46 on page 389.
`caretrieve.rexx`	• Displays web page coded in the HTML under the RETRIEVECONTENT subsection (of a TEMPLATE section). This HTML prompts the user to enter the transaction ID and a password if the user entered one when requesting the certificate. • When the user clicks **Retrieve and install certificate**, this CGI passes the transaction ID parameter to `cagetcert.rexx`.	See Figure 47 on page 390.
`cagetcert.rexx`	• Displays web page coded in the HTML under RETURNCERT subsection (of a TEMPLATE section). This HTML determines which of the following forms to use when returning the certificate: – as a base64-encoded certificate (for server certificates) – as an ActiveX object (for Microsoft Internet Explorer browser certificates) – as an application/x-x509-user-certificate MIME type (for Mozilla-based browser certificates).	See Figure 49 on page 392.
`cagetcert2.rexx`	• Displays web page coded in the HTML under RETURNCERT subsection (of a TEMPLATE section). This HTML returns the certificate and private key in PKCS #12 format.	
`cagorcvr.rexx`	• Displays web page coded in HTML under the FINDRECOVERCONTENT subsection of the APPLICATION section. This subsection displays security questions for users to answer. The answers to these questions can be used to recover a passphrase, which is used to recover a certificate.	

Table 41. CGI actions for end-user web pages (continued)

CGI exec	Action	Sample web page
`cadisplay.rexx`	• Displays web page coded in the HTML under the RECONTENT subsection (of the APPLICATION section). • For renewing a certificate, the user fills in the passphrase and clicks **Renew**. For revoking a certificate, the user clicks **Revoke**. Both actions call `camodify.rexx`.	See Figure 55 on page 397.
`camodify.rexx`	• Displays web page coded in the HTML under the SUCCESSCONTENT subsection (of a TEMPLATE section) for a successful renewal. The SUCCESSCONTENT subsection includes a **Continue** button the user can click to call `caretrieve.rexx`. • Displays the web page coded in HTML under the RESUCCESSCONTENT subsection (of the APPLICATION section) for a successful revocation.	See Figure 46 on page 389.
`installcert.rexx`	• When a user who is using the Internet Explorer browser clicks a link in a notification email for a renewed certificate, this CGI displays a web page where the user can paste the renewed certificate sent in the email and install it in the browser.	

Steps for performing minimal customization

You need to perform these steps only if you are customizing certificate templates for the first time. If your company used an earlier release of PKI Services, you do *not* need to do so again.

Before you begin
Review the certificate templates and decide if there are any that you want to remove from the `pkiserv.tmpl` certificates template file. If so, do this first. (To remove a certificate template, you can simply remove its name from the appropriate APPLICATION sections.)

Notes:

- Fields such as `%%Org%%` and `%%Country%%` are used to form the subject's distinguished name. Therefore, make sure that the name formed has a suffix that matches a suffix that the LDAP directory supports (that is, that it matches one of the suffix values in the LDAP server configuration file).
- The default name of the LDAP server configuration file is `ds.conf` for the IBM Tivoli Directory Server for z/OS LDAP server.

Perform the following steps to do the minimal updates on the remaining certificate templates:

1. For the SAF templates, update the following fields as needed:

   a. If present, replace the OrgUnit values in the following lines with values more appropriate to your organization:

   ```
   %%OrgUnit=Nuts and Bolts Division%%
   %%OrgUnit=SAF template certificate%%
   ```

   b. Replace `taca` in the following line with the correct label of the CERTAUTH signing certificate:

   ```
   %%SignWith=SAF:CERTAUTH/taca%%
   ```

   ____

2. For the PKI templates, replace the OrgUnit value in the following line with a value more appropriate for your organization:

```
%%OrgUnit=Class 1 Internet Certificate CA%%
```

____

3. If present, replace `The Firm` with the name of your company in the following `%%Org` line:

   ```
   %%Org=The Firm%%
   ```

   ____

4. If your company location is not the United States, update the following line by specifying the correct two-letter country abbreviation:

   ```
   %%Country=US%%
   ```

   ____

5. If present, replace `host-name` with the domain name of this system in the following `%%HostIdMap` line:

   ```
   %%HostIdMap=@host-name%%
   ```

   You also need to follow the instructions in "Administering HostIdMappings extensions" on page 482.

   ____

6. For non-SAF certificates, you can notify users when certificate requests are rejected or when certificates are ready for retrieval or are expiring.

   a. If you do not want to have NotifyEmail appear as an input field for any non-SAF certificates, delete the NotifyEmail lines in the following locations in the TEMPLATE section for this certificate:

      - In the header:

        ```
        # NotifyEmail - optional
        ```

      - In the list of fields:

        ```
        %%NotifyEmail (optional)%%
        ```

   b. If you do not want to have NotifyEmail appear as an input field for renewal of any non-SAF certificates, delete the following NotifyEmail line in the APPLICATION section and in the list of fields:

      ```
      %%NotifyEmail (optional)%%
      ```

   ____

7. Insert the copyright statement for your company in the `-copyright` named field in the INSERT section.

   ____

8. Insert the email address of your company's PKI Services administrator in the `-pagefooter` named field in the INSERT section.

   ____

Steps for additional first-time customization

You need to perform these steps only if you are customizing certificate templates for the first time. If your company used an earlier release of PKI Services, you do *not* need to perform these steps.

Perform the following steps if you want to perform additional customization of the end-user web pages:

1. Review the templates and decide which you need to update.

---

2. If necessary, change the true name, alias, or nickname, as in the following lines.

   ```
   <TEMPLATE NAME=true_name>
   <TEMPLATE NAME=alias>
   <NICKNAME=nickname>
   ```

   true_name
   Is the whole and complete name of the certificate template.

   alias
   Differentiates browser from server certificates. An alias is not required. You can have more than one alias.

   nickname
   Is an 8-character name. SAF certificates do not have nicknames. If a nickname is not present, the certificate is not automatically renewable.

   Example:

   ```
   <TEMPLATE NAME=1-Year PKI SSL Browser Certificate>
   <TEMPLATE NAME=PKI Browser Certificate>
   <NICKNAME=1YBSSL>
   ```

---

3. If necessary, in the CONTENT subsection, change the certificate fields listed. The following example is from the one-year PKI SSL browser certificate template.

   Example:

   ```
   <p> Enter values for the following field(s)
    %%CommonName%%
    %%Requestor (optional)%%
    %%PassPhrase%%
    %%PublicKey2[browsertype]%%
   ```

---

4. If you add required fields in the preceding step, update the JavaScript code that is part of the embedded HTML to check for required fields that are missing.

   Example:

   ```
   ValidCommonName(frm) &&
   ValidPassPhrase(frm) &&
   ValidPublicKey2(frm) &&
   ```

---

5. If necessary, in the APPL subsection, change the list of certificate fields that the application provides. (Currently, the only supported fields are `UserId` and `HostIdMap`.) The following example is from the two-year PKI browser certificate for authenticating to z/OS:

   Example:

   ```
   <APPL>
    %%UserId%%
    %%HostIdMap=@host-name%%
   </APPL>
   ```

---

6. If necessary, in the CONSTANT subsection, update the list of certificate fields whose values are hardcoded. The following example is from the one-year PKI SSL browser certificate template:

   Example:

   ```
   <CONSTANT>
    %%NotBefore=0%%
    %%NotAfter=365%%
   ```

```
 %%KeyUsage=handshake%%
 %%OrgUnit=Class 1 Internet Certificate CA%%
 %%Org=The Firm%%
 %%SignWith=PKI:%%
</CONSTANT>
```

Note: If you update the CONSTANT subsection to create subject distinguished names, make sure that the names match the LDAP suffix that are defined for your LDAP server. Otherwise, the certificates are not posted to LDAP. PKI Services constructs the subject distinguished name from the fields that are specified in the following order:

- CommonName
- Title
- OrgUnit (if repeating, in the order that they appear in the template file)
- Org
- Locality
- StateProv
- Country

____

7. If necessary, edit the ADMINAPPROVE subsection. (Certificates requiring an administrator's approval have an ADMINAPPROVE subsection. The absence of the ADMINAPPROVE subsection indicates that requests are automatically approved and that they can be synchronously fulfilled.) Make sure the ADMINAPPROVE subsection, if present, correctly lists the minimum set of certificate fields that the administrator can change.

   Note:

   a. There might be more fields in the ADMINAPPROVE subsection than fields that the user can complete in the certificate request (because the users do not necessarily see all fields).
   b. Do not include the `Requestor`, `Label`, `UserId`, `PublicKey`, or `SignWith` fields in the ADMINAPPROVE subsection. These fields cannot be changed and are ignored if present. (For a list of tags that are allowed in the ADMINAPPROVE subsection, see the subsection ADMINAPPROVE, in the topic about "TEMPLATE sections" on page 151.)
   c. If a request is examined and approved by more than one PKI Services administrator, include the ADMINNUM=*value* tag in the ADMINAPPROVE subsection.

   The following example of the ADMINAPPROVE subsection is from the one-year PKI SSL browser certificate template:

   Example:

```
<ADMINAPPROVE>
 %%CommonName (Optional)%%
 %%OrgUnit (Optional)%%
 %%OrgUnit (Optional)%%
 %%Org (Optional)%%
 %%NotBefore (optional)%%
 %%NotAfter (Optional)%%
 %%KeyUsage (Optional)%%
 %%HostIdMap (Optional)%%
 %%HostIdMap (Optional)%%
 %%HostIdMap (Optional)%%
 %%HostIdMap (Optional)%%
</ADMINAPPROVE>
```

   Note: The four `%%HostIdMap%%` lines in the example indicate that the approver can provide up to four `HostIdMap` entries.

____

8. If necessary, update the following subsections:

   - The SUCCESSCONTENT subsection contains only the `%%-requestok%%` named field, which contains the HTML for the web page whose main heading is "Request submitted successfully". To make changes to this web page, update the `-requestok` INSERT (in the INSERT section of `pkiserv.tmpl`):

```
<INSERT NAME=-requestok>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN""http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
<TITLE> Web Based Certificate Generation Success</TITLE>
</HEAD>
<BODY>
<H1> Request submitted Successfully</H1>
[errorinfo]
<p> Here's your transaction ID. You will need it to retrieve your
certificate. Press 'Continue' to retrieve the certificate.
<p> <TABLE BORDER><TR><TD>[transactionid]</TD></TR></TABLE>
<FORM METHOD=GET ACTION="/PKIServ/ssl-cgi/caretrieve.rexx">
<INPUT NAME="Template" TYPE="hidden" VALUE="[tmplname]">
<INPUT NAME="TransactionId" TYPE="hidden" VALUE="[transactionid]">
<INPUT TYPE="submit" VALUE="Continue">
</FORM>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>
```

- The FAILURECONTENT subsection contains only the `%%-requestbad%%` named field, which contains the HTML for the web page whose main heading is "Request was not successful". To make changes to this web page, update the `requestbad` INSERT:

```
<INSERT NAME=-requestbad>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN""http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
<TITLE> Web Based Certificate Generation Failure</TITLE>
</HEAD>
<BODY>
<H1> Request was not successful</H1>
<p> Please correct the problem or report the error to your Web admin
person<br>
<PRE>
[errorinfo]
</PRE>
<p>%%-pagefooter%%
</BODY>
</HTML>
</INSERT>
```

____

9. If necessary, update the RETRIEVECONTENT subsection.

Note: See "Steps for changing the runtime user ID for retrieving certificates" on page 228 for directions for changing the runtime user ID for retrieving a certificate.

   a. The RETRIEVECONTENT subsection includes the `%%-copyright%%` named field. If you want to make any changes in the copyright statement, update the copyright INSERT. (The following sample is the copyright INSERT as it is originally provided in the `pkiserv.tmpl` file. You should have previously updated this INSERT by providing information tailored to your company, as described in "Steps for performing minimal customization" on page 219.)

```
<INSERT NAME=-copyright>
 <!--
/*******************************************************************/
/*                                                                 */
/* LICENSED MATERIALS - PROPERTY OF IBM                            */
/* 5650-ZOS                                                        */
/* COPYRIGHT IBM CORP. 2001, 2019                                  */
/*                                                                 */
/*******************************************************************/
--> </INSERT>
```

   b. If necessary, update any web page content (such as headers, footers, titles, background colors, frames, links, and so on) for the web page whose main heading is "Retrieve Your (certificate template name)".

____

10. If you are updating the template for a server certificate, you can update the HTML in the RETURNCERT subsection to customize the returned web page. (For a browser template, you cannot change the RETURNCERT subsection. It must contain the `%%returnbrowsercert%%` named field,

which contains the [browsertype] substitution variable. The INSERT section contains browser-specific returnbrowsercert INSERTs.)

____

Steps for retrofitting release changes into the PKI Services certificate templates

If you used an earlier release of PKI Services, you might need to retrofit changes in the pkiserv.tmpl certificate templates file. (You would not want to replace the file if you customized it in the previous release.)

You can use a file comparison tool to compare the new PKI Services certificates template file (/usr/lpp/pkiserv/samples/pkiserv.tmpl) and your existing PKI Services certificates template file (/etc/pkiserv/pkiserv.tmpl).

Perform the following steps to retrofit changes into the pkiserv.tmpl certificate templates file so you do not lose any customization that you made in a previous release.

1. Make a backup copy of your current certificate templates file. For example, enter from the UNIX command line:

   ```
   cp /etc/pkiserv/pkiserv.tmpl /etc/pkiserv/pkiserv.backup
   ```

   ____

2. Copy the new sample templates file to the runtime location. (This is the copy you edit.)

   ```
   cp /usr/lpp/pkiserv/samples/pkiserv.tmpl /etc/pkiserv/pkiserv.tmpl
   ```

   ____

3. Using a compare program of your choice, compare the two template files:
   - /etc/pkiserv/pkiserv.tmpl
   - /etc/pkiserv/pkiserv.backup

   ____

4. Edit the runtime copy of the templates file (/etc/pkiserv/pkiserv.tmpl). Using the compare output that is generated in Step 3, merge the changes that you made to the original template file into the runtime copy of the templates file.

   ____

5. Exit the file to save your changes.

   ____

Locating code for customizing end-user web pages

For ongoing customization of end-user web pages, you must know the code locations for those web pages. The following table summarizes this information:

Table 42. Location of code for various web pages

Main header (and sample web page if any)	Location of code in pkiserv.tmpl certificate templates file
"1-Year PKI S/MIME Browser Certificate"	TEMPLATE section, CONTENT subsection
"1-Year PKI SSL Browser Certificate" (See Figure 44 on page 386.)	TEMPLATE section, CONTENT subsection

Table 42. Location of code for various web pages (continued)

Main header (and sample web page if any)	**Location of code in pkiserv.tmpl certificate templates file**
"2-Year EV SSL server certificate"	TEMPLATE section, CONTENT subsection
"2-Year PKI Browser Certificate For Authenticating To z/OS"	TEMPLATE section, CONTENT subsection
"2-Year PKI Authenticode - Code Signing Certificate"	TEMPLATE section, CONTENT subsection
"5-Year PKI Intermediate CA Certificate"	TEMPLATE section, CONTENT subsection
"5-Year PKI IPSEC Server (Firewall) Certificate"	TEMPLATE section, CONTENT subsection
"5-Year PKI SSL Server Certificate"	TEMPLATE section, CONTENT subsection
"5-Year SCEP certificate - Preregistration"	TEMPLATE section, CONTENT subsection
"2-Year EST Certificate - Preregistration"	TEMPLATE section, CONTENT subsection
"*n*-Year PKI Certificate for Extensions Demonstration"	TEMPLATE section, CONTENT subsection
"Here's Your Certificate. Cut and Paste it to a File"	INSERT section, `-return10cert` INSERT. (This is referenced in the RETURNCERT subsection of the TEMPLATE section of each certificate template.)
"Internet Explorer Certificate Install" (See Figure 49 on page 392.)	INSERT section, `returnbrowsercertIE` INSERT
"Preregistration successful"	INSERT section, `-preregok` INSERT. (This is referenced in the SUCCESSCONTENT subsection of the TEMPLATE section of each certificate template.)
"PKI Services Certificate Generation Application" (See "#unique_66/unique_66_Connect_42_mainpage" on page 376.)	APPLICATION section, CONTENT subsection
"Renew or Revoke a Browser Certificate" (See Figure 55 on page 397.)	APPLICATION section, RECONTENT subsection
"Internet Explorer Auto Renewed Certificate Install"	APPLICATION section, RENEWEDCERT subsection
"Request submitted successfully" (For submitting a successful certificate request or renewal, see Figure 46 on page 389.)	• For a successful certificate request or renewal: INSERT section, `-requestok` INSERT. (This is referenced in the SUCCESSCONTENT subsection of the TEMPLATE section of the appropriate certificate template.) • For a successful certificate revocation: INSERT section, `-renewrevokeok` INSERT. (This is referenced in the RESUCCESSCONTENT subsection of the APPLICATION section.)

Table 42. Location of code for various web pages (continued)

Main header (and sample web page if any)	Location of code in pkiserv.tmpl certificate templates file
"Request was not successful"	• For an unsuccessful certificate request: INSERT section, `-requestbad` INSERT. (This is referenced in the FAILURECONTENT subsection of the TEMPLATE section of each certificate template.) • For an unsuccessful certificate revocation request: INSERT section, `-renewrevokebad` INSERT. (This is referenced in the REFAILURECONTENT subsection of the APPLICATION section.)
"Retrieve Your 1-Year PKI S/MIME Browser Certificate"	TEMPLATE section, RETRIEVECONTENT subsection
"Retrieve Your 1-Year PKI SSL Browser Certificate" (See Figure 47 on page 390.)	TEMPLATE section, RETRIEVECONTENT subsection
"Retrieve Your 2-Year PKI Browser Certificate For Authenticating To z/OS"	TEMPLATE section, RETRIEVECONTENT subsection
"Retrieve Your 2-Year PKI Authenticode - Code Signing Certificate"	TEMPLATE section, RETRIEVECONTENT subsection
"Retrieve Your 5-Year PKI Intermediate CA Certificate"	TEMPLATE section RETRIEVECONTENT subsection
"Retrieve Your 5-Year PKI IPSEC Server (Firewall) Certificate"	TEMPLATE section, RETRIEVECONTENT subsection
"Retrieve Your 5-Year PKI SSL Server Certificate"	TEMPLATE section, RETRIEVECONTENT subsection
"Retrieve Your `[tmplname]`"	TEMPLATE section, RETRIEVECONTENT subsection
"Retrieve Your SAF Browser Certificate 1-Year"	TEMPLATE section, RETRIEVECONTENT subsection
"Retrieve Your SAF Server Certificate 1-Year"	TEMPLATE section, RETRIEVECONTENT subsection
"SAF Browser Certificate 1-Year (Auto Approved)"	TEMPLATE section, CONTENT subsection
"SAF Server Certificate 1-Year (Auto Approved)"	TEMPLATE section, CONTENT subsection

Note: Fields (such as the Key Usage (`KeyUsage`) drop down or the Organizational Unit (`OrgUnit`) text field) are defined in the `pkiserv.tmpl` certificate templates file, in the INSERT section. (See Table 32 on page 139 for descriptions of the fields.)

Steps for adding a new certificate template

Perform the following steps to add a new certificate template:

1. Review the contents of the certificate templates provided with PKI Services to determine the one that most closely approximates the certificate template you want to add.

   ____

2. After you determine the certificate template to use as a model, copy the section defining the model template in the certificate templates file.

   ____

3. Provide a new name, alias, and, if present, nickname for the certificate template.

   ____

4. Follow the steps for customizing certificate templates, starting at Step "3" on page 221.

   ____

Changing the runtime user ID

When the PKI Services CGIs are called, they are assigned a runtime user ID. This is the identity that is associated with the unit of work (task). This identity must be authorized to call the function being requested. (See Chapter 22, "RACF administration for PKI Services," on page 481 for more information.) Most of the templates run under the surrogate user ID (PKISERV) for requesting a certificate and for then retrieving it.

There are two exceptions:

- The two SAF templates run under PKISERV for requesting a certificate but run under the client's user ID for certificate retrieval.
- The five-year PKI intermediate CA template runs under the client's user ID for requesting a certificate and for certificate retrieval.

The advantage of having PKISERV as the runtime user ID is that this is the only user ID that needs to be authorized for requesting certificates. The advantage of using the client's user ID is that you have greater control over who can request and retrieve certificates. For example, you can require the user to authenticate by entering user ID and password before requesting or retrieving a certificate.

You can control the user ID under which a certificate request or retrieval runs by selectively commenting and uncommenting FORM statements in the `pkiserv.tmpl` file. (For requesting a certificate, the FORM statements are in the appropriate TEMPLATE section, in the CONTENT subsection. For retrieving a certificate, the FORM statements are in the appropriate TEMPLATE section, in the RETRIEVECONTENT subsection.)

There are three levels of access control for requesting and retrieving certificates:

- Under the client's ID with user ID and password authentication
- Under the surrogate user ID with user ID and password authentication
- Under the surrogate user ID without user ID and password authentication.

The IBM HTTP Server configuration file enforces these three levels of access control. The default configuration for PKI Services maps the three levels of access control to the following CGI directories:

- `/PKIServ/ssl-cgi-bin/auth`
- `/PKIServ/ssl-cgi-bin/surrogateauth`
- `/PKIServ/ssl-cgi-bin`

Each of the request and retrieve CGIs is in all three directories. Thus, when you run a CGI you get the protection established for the directory from which it is called.

Each certificate template contains several FORM statements (two commented out and one uncommented, which is active) that determines which of these apply. You can change the access control by uncommenting one of the FORM statements that is commented out and commenting out the one that is active.

Steps for changing the runtime user ID for requesting certificates

Perform the following steps to change the runtime user ID for requesting a certificate.

1. In the `pkiserv.tmpl` file, find the CONTENT subsection of the TEMPLATE section for the template whose user ID you want to change. Locate the lines containing the FORM statements, such as those in the following example:

   Example:

   ```
   <h3><li>Request a New Certificate
   # This ACTION forces userid/pw authentication and runs the task under
   # the client's ID
   #<FORM NAME="CertReq" METHOD=POST ACTION=
   #                "/PKIServ/ssl-cgi-bin/auth/careq.rexx" onSubmit=

   # This ACTION forces userid/pw authentication but runs the task under
   # the surrogate ID
   #<FORM NAME="CertReq" METHOD=POST ACTION=
   #              "/PKIServ/ssl-cgi-bin/surrogateauth/careq.rexx" onSubmit=

   # This ACTION is for non z/OS clients. The task runs under the
   # surrogate ID
    <FORM NAME="CertReq" METHOD=POST ACTION=
                  "/PKIServ/ssl-cgi-bin/careq.rexx" onSubmit=
   ```

   Notice that the preceding lines contain three FORM statements. The first two FORM statements are commented out, so they are not active. They are for:

   - Requesting the certificate under the client's ID and using user ID and password authentication
   - Requesting the certificate under the surrogate ID and using user ID and password authentication

   The third FORM statement is for requesting the certificate under the surrogate user ID without user ID and password authentication. This is active (it is not commented out).

   ____

2. To change the runtime user ID, remove the comment delimiter (#) from in front of the lines for the commented-out FORM statement you want to use and insert the comment delimiter in front of the lines for the end FORM statement.

   ____

Steps for changing the runtime user ID for retrieving certificates

Perform the following steps to change the runtime user ID for retrieving a certificate.

1. In the `pkiserv.tmpl` file, find the RETRIEVECONTENT subsection of the TEMPLATE section for the template whose user ID you want to change. Locate the lines containing the FORM statements, such as those in the following example:

   Example:

   ```
   <H1> Retrieve Your [tmplname]
   <H3>Please bookmark this page
   <p>Since your certificate may not have been issued yet, we recommend
   that you create a bookmark to this location so that when you return to
   this bookmark, the browser will display your transaction ID.
   This is the easiest way to check your status.

   # This ACTION forces userid/pw authentication and runs the task
   # under the client's ID
   #<FORM NAME=retrieveform METHOD=POST ACTION=
   #      "/PKIServ/ssl-cgi-bin/auth/cagetcert.rexx" onSubmit=
   #
   # This ACTION forces userid/pw authentication but runs the task
   # under the surrogate ID
   #<FORM NAME=retrieveform METHOD=POST ACTION=
   #      "/PKIServ/ssl-cgi-bin/surrogateauth/cagetcert.rexx" onSubmit=
   #
   # This ACTION is for non z/OS clients. The task runs under surrogate ID
    <FORM NAME=retrieveform METHOD=POST ACTION=
         "/PKIServ/ssl-cgi-bin/cagetcert.rexx" onSubmit=
   ```

   Notice that the preceding lines contain three FORM statements. The first two FORM statements are commented out (they are not active). These are for:

- Retrieving the certificate under the client's ID
- Retrieving it under the surrogate ID, but requiring user ID and password authentication.

The third FORM statement is for retrieving the certificate under the surrogate user ID without user ID and password authentication. This is active (it is not commented out).

____

2. To change the runtime user ID, remove the comment delimiter (#) from in front of the lines for the commented-out FORM statement you want to use and insert the comment delimiter in front of the lines for the end FORM statement.

____

Customizing the OtherName field

When you use the OtherName field, you are able to bind additional identities or owner information to the subject of the certificate using the subject alternate name extension. These identities might take different forms, such as employee numbers, customer account numbers, and other identities that you choose to use.

The OtherName value is a concatenated string that consists of one or more pairs of OIDs and their associated values. The string is saved in the subject alternate name extension in the certificate.

PKI Services implements the OtherName field as a customizable INSERT called `AltOther_<OID>`. The following certificate template in `pkiserv.tmpl` is supplied to illustrate the use of the INSERT fields.

```
Template Name - n-Year PKI Certificate for Extensions Demonstration
```

The *n*-year PKI certificate template builds a certificate using information provided primarily by users, rather than information that you control. For demonstration purposes, the template builds a certificate that contains all extensions that are supported by PKI Services. The template contains two sample OtherName fields:

%%AltOther_1_2_3_4_5%%
Builds one input field.

%%AltOther_1_2_3_4_6%%
Builds two input fields.

The `AltOther_1_2_3_4_5` string represents an OtherName field with OID `1.2.3.4.5`, an 11-character string that stores a customer account number. The `AltOther_1_2_3_4_6` string represents an OtherName field with OID `1.2.3.4.6`, a 17-character string that stores a 9-digit license number and an expiration date in the *yyyymmdd* format.

When you choose to use the OtherName field to build the subject alternate name extension, you might also want to customize the end-user web pages to allow end-users to enter the required information using customized input screens that are easier for them to use. Rather than asking a user to enter a string, as shown in the following example, you can prompt the user to enter a 9-digit license number and its expiration date.

Example of an OtherName field value:

```
1.2.3.4.6,12345678920050215
```

Steps for customizing the sample AltOther_*<OID>* INSERTs

Before you begin

- Decide what identifiers you want to add to the Subject Name Alternate extension.
- Select the registered OID value to use to represent your data string. Check the appropriate standards organization (ISO or ITU). If not already registered, register your own OID.

- Select which certificate templates you want to update to add the Subject Name Alternate extension.
- Decide whether to use a sample INSERT for your `AltOther_<OID>` INSERT or create your own INSERT. The sample INSERT called `AltOther_1_2_3_4_5` demonstrates using one input field. The sample insert that is called `AltOther_1_2_3_4_6` demonstrates using two input fields.
- Determine the following values that you want to use to customize your INSERT.
  - The OID value for your OtherName field
  - The name and length for each input field.
- Review Figure 3 on page 230. It contains a listing of the sample INSERT called `AltOther_1_2_3_4_6` which demonstrates using two input fields. The lines you are most likely to customize are marked in Figure 3 on page 230. The following steps refer to the marked lines.

Procedure

Perform the following steps to customize the `AltOther_<OID>` INSERT using Figure 3 on page 230 as a reference.

```
  ===================================================================
# Sample AltOther INSERT with two input fields
# ===================================================================
<INSERT NAME=AltOther_1_2_3_4_6> 1
<INPUT NAME="AltOther_1_2_3_4_6" TYPE="hidden" maxlength="255">

<p> Other Name for alternate name: <BR>
<p> <LABEL for="other2afield">Customer's driver license number (9 digits) [optfield]</LABEL> <BR> 2
<INPUT NAME="Other2a" TYPE="text" SIZE=9 maxlength="9" [readonly]     id="other2afield"> 3
<p> <LABEL for="other2bfield">Customer's driver license expiration date (yyyymmdd) [optfield]</LABEL> <BR> 4
<INPUT NAME="Other2b" TYPE="text" SIZE=8 maxlength="8" [readonly]     id="other2bfield"> 5

<INPUT NAME="altrawstring_1_2_3_4_6" TYPE="hidden" VALUE="[altrawvalue]">

<SCRIPT LANGUAGE="JavaScript">
<!--
//This is the script that will be called at load time.
var form=document.forms[0]
if (form.altrawstring_1_2_3_4_6.value.length > 0) {
  //The name 'Otherx' needs to match with the above INPUT NAME.
  //Substr(start position, length)
  form.Other2a.value=form.altrawstring_1_2_3_4_6.value.substr(0,9) 6
  form.Other2b.value=form.altrawstring_1_2_3_4_6.value.substr(9,8) 7
}
//-->
</SCRIPT>
<SCRIPT LANGUAGE="JavaScript"> 8
<!--
//This is the validation script
function ValidAltOther_1_2_3_4_6(frm){
 if (("[optfield]" == "" && frm.Other2a.value.length != 9) ||
     ("[optfield]" !="" && frm.Other2a.value != "" && frm.Other2a.value.length != 9)) {
  alert("Enter 9 digit license number.");
  frm.Other2a.focus();
  return false;
 }
:
 //Build the entire AltOther field.
 if (frm.Other2a.value != "" && frm.Other2b.value != "")
  frm.AltOther_1_2_3_4_6.value = "1.2.3.4.6," + frm.Other2a.value + 9
                     frm.Other2b.value;
 else
  frm.AltOther_1_2_3_4_6.value = "";
 return true;
}
//-->
</SCRIPT>
</INSERT>
```

Figure 3. Partial listing of the AltOther_1_2_3_4_6 sample INSERT showing the lines you are most likely to customize

1. Change the OID value `_1_2_3_4_6` to the OID value you need in the line marked 1 and in all other lines in the sample INSERT. For example, if you chose OID `2.16.76.1.3.1` for your OtherName field, change all occurrences of `AltOther_1_2_3_4_6` to `AltOther_2_16_76_1_3_1`.

   ____

2. Customize the first input field description in the line marked 2 to prompt users of your web page. For example, change `Customer's driver license number (9 digits)` to `Enter your member card number`.

   ____

3. Customize the first INPUT field name "`Other2a`" to your value in the line marked 3 and in all other lines in the sample INSERT. For example, change all occurrences of "`Other2a`" to "`MemNum`". Also, customize SIZE and `maxlength` as needed.

   ____

4. Customize the next input field description in the line marked 4 to prompt users of your web page. For example, change `Customer's driver license expiration date (yyyymmdd)` to `Enter your date of birth (yyyymmdd)`.

   ____

5. Customize the next INPUT field name "`Other2b`" to your value in the line marked 5 and in all other lines in the sample INSERT. For example, change all occurrences of "`Other2b`" to "`Birthdate`". Also, customize SIZE and `maxlength` as needed.

   ____

6. Customize the starting positions and lengths for each input field value in the lines marked 6 and 7. For example, if the member card number is an 11-digit number, change `form.Other2a.value=form.altrawstring_1_2_3_4_6.value.substr(0,9)` to `form.MemNum.value=form.altrawstring_2_16_76_1_3_1.value.substr(0,11)`.

   ____

7. Customize the validation script that begins with the line marked 8.

   ____

8. Change the OID value `1.2.3.4.6` to the OID value you need in the line marked 9. For example, if you chose OID `2.16.76.1.3.1` for your OtherName field, change `1.2.3.4.6` to `2.16.76.1.3.1`.

   ____

9. Repeat steps "2" on page 231 through "8" on page 231 for each additional input field you need.

   ____

Chapter 12. Customizing the administration web pages if you use REXX CGI execs

This information applies if you are using REXX CGI execs for your PKI Services web pages. If you are using JavaServer pages (JSPs), see Chapter 13, "Implementing the web application using JavaServer pages," on page 237.

CGIs for administration web pages

CGIs for administration web pages are execs that gain control when the user clicks an action button and render the web pages dynamically. All of the administration CGIs are contained in the `/usr/lpp/pkiserv/PKIServ/ssl-cgi-bin/auth` directory.

Table 43 on page 233 (which lists the CGI execs in logical order) summarizes the actions that they perform:

Table 43. CGI actions for administrative web pages

CGI exec	Action	Sample web page
`admmain.rexx`	This displays the administration home page. The main heading is "PKI Services Administration". This web page lets the administrator work with a single certificate request or certificate or search for certificate requests or certificates.	See Figure 68 on page 411.
`admpend.rexx`	On the administration home page, the administrator can search for certificate requests. The result of this search is one of the following web pages: • "Certificate Requests" web page. This web page lists certificate requests matching the criteria and allows the administrator to process certificate requests. • "Processing was not successful" web page.	For an example of the "Certificate Requests" web page, see Figure 73 on page 418.
`admpendtid.rexx`	On the administration home page, the administrator can enter a transaction ID to work with a single certificate request. This displays a web page whose main heading is one of the following: • "Single Request" - This lists the certificate request that matches the transaction ID and allows the administrator to process that certificate request. • "Processing was not successful"	For an example of the "Single Request" web page, see Figure 69 on page 412.
`admmodtid.rexx`	This displays the "Modify and Approve Request" web page that appears when the administrator decides to modify a request before approving it (on the "Single Request" web page).	See Figure 72 on page 415.

Table 43. CGI actions for administrative web pages (continued)

CGI exec	Action	Sample web page
`admicl.rexx`	On the administration home page, the administrator can search for certificates. This displays a web page whose main heading is one of the following: • "Issued Certificates" - This lists the certificate or certificates that match the search criteria. It also allows the administrator to revoke or delete selected certificates. • "Processing was not successful"	For a sample of the "Issued Certificates" web page, see Figure 73 on page 418.
`admiclcert.rexx`	On the administration home page, the administrator can enter a serial number to work with a single certificate. This displays a web page whose main heading is one of the following: • "Single Issued Certificate" - This lists the certificate that matches the serial number ID and allows the administrator to revoke or delete that certificate. • "Processing was not successful"	For a sample of the "Single Issued Certificate" web page, see Figure 77 on page 424.
`admacttid.rexx`	Displays a web page after the administrator processes a single certificate request (approving it with or without modifications, rejecting, or deleting it). This web page has one of the following as its main heading: • "Processing successful" • "Processing was not successful"	For a sample of the web page whose main heading is "Processing successful", see Figure 70 on page 414.
`admacttid2.rexx`	This displays a web page after the administrator approves a certificate request with modifications. The web page has one of the following main headings: • "Processing successful" • "Processing was not successful"	For a sample of the web page whose main heading is "Processing successful", see Figure 70 on page 414.
`admpendall.rexx`	After the administrator searches for certificate requests and `admpend.rexx` displays the results, the administrator clicks a button to approve, reject, or delete selected certificate requests. This calls admpendall.rexx, whose main heading is one of the following: • "Processing successful" if the action was successful • "Processing was not successful" if the action failed (for example, if the administrator tried to delete certificate requests that were already deleted) • "Processing partially successful" if not all of the selected requests are processed successfully	• For an example of the "Processing successful" web page, see Figure 74 on page 420. • For an example of the "Processing was not successful" web page, see Figure 75 on page 421. • For an example of the "Processing partially successful" web page, see Figure 76 on page 421.

Table 43. CGI actions for administrative web pages (continued)

CGI exec	Action	Sample web page
`admactcert.rexx`	Displays a web page after the administrator tries to revoke or delete one or more selected certificates. The web page has one of the following main headings: • "Processing successful" • "Processing was not successful"	-
`admiclall.rexx`	After the administrator searches for certificates and `admicl.rexx` displays the results, the administrator clicks a button to revoke or delete selected certificates. This calls `admiclall.rexx`, which displays a web page whose main heading is one of the following: • "Processing successful" if the action was successful • "Processing was not successful" if the action failed • "Processing partially successful" if not all of the selected certificates are processed successfully	-

Customizing the administration web pages

The administration web pages are not as customizable as the end-user web pages. You can customize page headers, footers, frames, links, colors, and so forth, but you cannot change internal web page content. Except for identifying the fields that an administrator can change when approving certificate requests, the administration web page logic is fixed.

However, you can make changes in the following subsections in the PKISERV APPLICATION section of the `pkiserv.tmpl` certificate template file. (These subsections appear in the application section of PKISERV only.)

ADMINHEADER
Contains the general installation-specific HTML content for the header of all the administration web pages.

ADMINFOOTER
Contains the general installation-specific HTML content for the footer of all the administration pages.

ADMINSCOPE
This optional subsection allows the administrator to choose a different CA domain. For more information, see “Adding a new CA domain” on page 302.

Steps for customizing the administration web pages

Perform the following steps to customize the administration web pages:

1. Add any web page header for the administration pages to the ADMINHEADER subsection of the PKISERV APPLICATION section. (The ADMINHEADER subsection is near the end of the APPLICATION section.)

   Example:

```
<ADMINHEADER>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN""http://www.w3.org/TR/html4/loose.dtd">
<HTML lang="en"><HEAD>
<TITLE>Web-Based Certificate Generation Administration</TITLE></HEAD>
<BODY>
</ADMINHEADER>
```

---

2. Add any web page footer for the administration pages to the ADMINFOOTER subsection of the APPLICATION section. (The ADMINFOOTER subsection is near the end of the APPLICATION section.)

   Example:

   ```
   <ADMINFOOTER>
   <p> email: webmaster@company.com
   </BODY>
   </HTML>
   </ADMINFOOTER>
   ```

Changing the runtime behavior for accessing administration pages

When the administrator tries to access the administration pages (by clicking the **Go to administration page** button on the PKI Services home page), access to the administration pages is controlled in one of the following ways:

- A popup window appears, requiring the administrator to enter a user name and password. (See Figure 67 on page 408 for a sample of the authentication popup window.)
- Alternately, the administrator might have to authenticate by using a previously issued browser certificate. In other words, the administrator would need to have a certificate before going to the administration web pages.

By default, the first method is used. However, you can change the runtime behavior so that the second method is used instead. If you decide to use the second method, anyone intending to become a PKI Services administrator needs to request and retrieve a one-year PKI browser certificate for authenticating to z/OS before trying to access the administration pages.

Note: The one-year PKI browser certificate for authenticating to z/OS contains a `HostIdMappings` extension. (For more information, see Chapter 22, “RACF administration for PKI Services,” on page 481.)

Steps for changing control of access to administration pages

Perform the following steps to change the access control of the administration pages to require authenticating by using a certificate:

1. Edit the `pkiserv.tmpl` certificate templates file and find the following lines in the PKISERV APPLICATION section:

   ```
   # The following action will force userid/pw authentication for administrators
   <FORM name=admform METHOD=GET ACTION="/PKIServ/ssl-cgi/auth/admmain.rexx">
   # The following action will force client certificate authentication
   # for administrators
   #<FORM name=admform METHOD=GET
   # ACTION="/PKIServ/clientauth-cgi/auth/admmain.rexx">
   <p>
   <INPUT TYPE="submit" VALUE="Go to Admin Pages">
   </FORM>
   ```

   The first FORM statement in these lines is active. (It is not commented out with # characters in front of the lines.) This requires authentication by entering the user name and password in a popup window. The second FORM statement is commented out (using # characters). This requires authentication by using a previously issued browser certificate.

---

2. Comment out the first FORM statement (add # characters in front of the FORM and ACTION lines) and uncomment the second FORM statement (removing the # characters in front of the FORM and ACTION lines).

Chapter 13. Implementing the web application using JavaServer pages

As an alternative to the REXX CGI execs and text template file described in Chapter 11, "Customizing the end-user web application if you use REXX CGI execs," on page 135 and Chapter 12, "Customizing the administration web pages if you use REXX CGI execs," on page 233, you can use JavaServer pages (JSPs) and an XML template file to create and customize the PKI Services web application. This approach has several advantages over the REXX CGI approach:

- It uses Java, a popular and flexible web application programming language.
- It uses XML, which is likely to be more familiar and intuitive to web application programmers than the text template file format used by the REXX CGI approach.
- You can validate the syntax of your XML template file using the TemplateTool utility (see "Using the TemplateTool utility" on page 442) or web application tools that include XML validation, such as IBM Rational® Software Development Platform.
- The XML template with modifiable JSP files is a simpler structure than the single text template file used by the REXX CGI approach.

JavaServer pages control the content or appearance of web pages through the use of Java code that runs on an application server to modify a web page before it is sent to the user who requested it. JavaServer pages can contain a mixture of HTML and Java code. For the PKI Services web application, the XML template file defines the applications and the certificates that the applications provide. The JavaServer pages define and process the web pages. The application server is assumed to be WebSphere Application Server.

Certificate templates files used with JSPs

When you implement the web application using JSPs, there are two versions of the certificate templates file:

- An XML templates file, `pkitmpl.xml`, used to customize the web application
- A text CGI templates file, `pkixgen.tmpl`, used by the PKI Services daemon

Whenever you update `pkitmpl.xml`, you must use the TemplateTool utility to create an equivalent copy of `pkixgen.tmpl`. For more information, see "Using the TemplateTool utility" on page 442. If you do not create an updated `pkixgen.tmpl` file, the daemon writes the following message to the daemon log file when it determines that `pkixgen.tmpl` is not current:

```
IKYC068I The templates file used may not be current
```

The default location of the two versions of the template file is `/etc/pkiserv/`. Alternatively, if the environment variable PKISERV_CONFIG_PATH is defined, PKI JSP processing uses the value of PKISERV_CONFIG_PATH as the location of `pkitmpl.xml` and `pkixgen.tmpl` . The XML schema is defined in the file `PKIServ.xsd` in the same directory as `pkitmpl.xml`.

Examining the pkitmpl.xml file

The PKI Services XML template, `pkitmpl.xml`, defines the following elements in the order defined by the PKIServ.xsd schema file:

- The PKI Services ActiveX install URL.
- One or more applications.

An application is a grouping of certificate request templates. This grouping might be done because the templates are shared among a set of end users or because the templates have a common administrator. A PKI Services installation can have one or more applications defined.

- One or more certificate request templates.

  A certificate request template is a predefined set of characteristics for certificate requests and the resulting certificates. Each certificate request template defines:

  - The type of certificate request (for example, browser or server, SCEP preregistration request, EST preregistration request, SAF or PKI certificate)
  - A name for the certificate request template, which is displayed to both the end user and the administrator
  - A nickname (maximum of 8 characters) that uniquely identifies each template, which PKI Services uses to retrieve information about the template
  - The values that you want a user to input and whether those values are optional or required
  - The values that you want to supply for the user
  - The values an administrator is allowed to supply
  - Whether the certificate is automatically approved
  - Whether the certificate is automatically renewed
  - A set of preregistration rules (applicable only to SCEP preregistration requests), which define whether approval is needed based on the level of authentication provided
  - EST preregistration requests have no preregistration rules to define since it is always AutoApprove.

The XML template file, `pkitmpl.xml`, begins by defining the PKI Services ActiveX install URLs.

```
<tns:CEnroll_install_URL>https://hostname:port/PKIServ/PKICEnroll/
PKICEnrollDeploy.msi<tns:CEnroll_install_URL>
```

The tag for `CEnroll_install_URL` specifies the URL of the PKI Services ActiveX control installer program. Each URL can be relative to the PKI Services web root context or absolute. The text "Install the PKI ActiveX Control to renew certificates" on the PKI Services home page for end users (see "#unique_66/unique_66_Connect_42_mainpage" on page 376) links to the URL specified for the version of Microsoft Windows running on the system. Users need to install the PKI Services ActiveX control to install renewed certificates using the Internet Explorer browser.

The application tag (`<tns:application>`) defines a particular set of end users. The application tag consists of an application name (in this case Customers) and one or more application templates (`<tns:appltemplate>`).

```
<tns:applname>Customers</tns:applname>
<tns:appltemplate>1-Year PKI SSL Browser Certificate</tns:appltemplate>
```

The contents of the `appltemplate` tag ("1-Year PKI SSL Browser Certificate", for example) corresponds to the `certname` element of a `certreq_template` tag.

A certificate request template is defined by a `certreq_template` tag, shown in Figure 4 on page 238.

```
<tns:certreq_template>
     <tns:certname>1-Year PKI SSL Browser Certificate </tns:certname>
     <tns:certtype>PKI Browser Certificate</tns:certtype>
     <tns:certtype_description>PKI Browser Certificate for Secured Connections</tns:certtype_description>
     ⋮
```

Figure 4. A `certreq_template` tag

The certificate type tag (`<tns:certtype>`) can have one of the following values:

- `PKI Preregistration` (for SCEP and EST preregistration requests)
- `PKI Browser Certificate`
- `PKI Server Certificate`

- `PKI Key Certificate` (for a certificate for which PKI Services generated the key pair)
- `SAF Browser Certificate`
- `SAF Server Certificate`

The certificate type description (`<tns:certtype_description>`) is an optional tag. Its contents are used on the web pages wherever the certificate type is to be displayed. This tag allows administrators to use words that they feel might be more understandable to their end users than the pre-defined values for the certificate types. A common use of this tag might be to translate the certificate types to another language. If this tag is omitted the contents of the certificate type tag (`<tns:certtype>`) is used as the certificate type description.

The request authentication type tag (`<tns:request_authtype>`) and retrieve authentication type tag (`<tns:retrieve_authtype>`) >) define the type of authentication that must be used to request or retrieve a certificate. The acceptable values for these tags are:

`noAuthRunAsSurrogate`
No authentication should be used. The task runs as a surrogate user.

`zAuthRunAsSurrogate`
The user is prompted to authenticate (log in) to z/OS using a RACF user ID and password. The task runs as a surrogate user.

`zAuthRunAsClient`
The user is prompted to authenticate (log in) to z/OS using a RACF user ID and password. The task runs as the client.

The next element of a certificate request template is the Auto-Approve indicator (`<tns:AutoApprove>`) A value of `Y` or `y` indicates that any certificate requests made with this template should be automatically approved (no administrator approval is required and the administrator does not have an opportunity to modify or reject certificate requests). A value of `N` or `n` indicates that certificate requests made with this template are not automatically approved and must be approved by an administrator.

The next element following the Auto-Approve indicator is the Synchronous indicator (<tns:Synchronous>). A value of *Y* or *y* indicates that any certificate requests made with this template should be synchonously created. A value of *N* or *n* indicates that certificate requests made with this template are not synchronously created.

The next element of a certificate request template is the Admin-Num indicator (`<tns:AdminNum>`). The value is to set the number of administrators that are required to approve a certificate request. If both `<tns:AdminNum>` and `<tns:AutoApprove>` Y exist, then `<tns:AdminNum>` takes precedence. Any improper values are handled the same way as the ADMINNUM entry in the CGI templates file.

Note: A request created from this template remains in Pending Approval state until the required number of individual administrative approvals is made for the request, at which time the request changes to Approved state. If an administrator issues an Approve with Modifications on a request that is in Pending Approval state, any previously made approvals are nullified, and the number of approvals that are made for the request is reset to 1.

The next element of a certificate request template is the Auto-Renew indicator (`<tns:AutoRenew>`). A value of `Y` or `y` indicates that any certificate created using this template is automatically renewed. A value of `N` or `n` indicates that certificates created using this template are not automatically renewed.

The following form fields are defined with tags in the certificate request template:

- AltDomain
- AltEmail
- AltIPAddr
- AltOther
- AltURI
- BusinessCat
- ClientName

- CommonName
- Country
- CustomExt
- DomainName
- DNQualifier
- EmailAddr
- ExtKeyUsage
- HostIdMap
- JurCountry
- JurLocality
- JurStateProv
- KeySize
- KeyUsage
- Label
- Locality
- Mail
- NotAfter
- NotBefore
- NotifyEmail
- Org
- OrgUnit
- PassPhrase
- PostalCode
- PublicKey
- Requestor
- Security
- SerialNumber
- SignWith
- StateProv
- Street
- Title
- Uid
- UnstructAddr
- UnstructName
- UserId

A form field tag has the form:

```
<tns:Name formtype="InstallationSpecified | AdminSpecified | UserSpecified"
                  initvalue="xxxxx"
                  optional=”true | false”
                  JSPfilename=”xxx.jsp”/>
```

where

Name
is the name of the form field, for example AltOther or Security.

formtype
can have one of the following values:

UserSpecified
The form field appears on the certificate request web page where the user can enter data.

InstallationSpecified
The value is provided by the XML template and is not displayed on the certificate request web page. Instead, there is a hidden form field on the certificate request web page that specifies the value.

AdminSpecified
The administration Approve with Modification web page should always display this form field and allow an administrator to specify a value for it.

The default for `formtype` is `UserSpecified`.

initvalue
The initial value of the form field. If `formtype` is `AdminSpecified` or `UserSpecified`, the form field is displayed with this initial value set but modifiable. If `formtype` is `Installation Specified`, this initial value is given to the hidden form field and it cannot be changed for the certificate request.

optional
Indicates whether this form field is optional. For `UserSpecified` form fields, `optional` indicates whether a value must be provided on the Certificate Request web page. For `AdminSpecified` form fields, `optional` indicates whether a value must be provided on the administrator's Approve with Modifications web page. For `InstallationSpecified` form fields, the optional attribute is ignored. The default value for `optional` is false, and the form field is required.

JSPfilename
The file name of a JSP file that is included to display and validate the form field. The file that is included is a modifiable include file in the `mod_inc` directory with your web application's EAR file. If `formtype` is `Installation Specified`, `JSPfilename` is ignored.

The default value for `JSPfilename` is the name of the form field, in lowercase, combined with the `.jsp` extension. For example, the default value for `JSPfilename` for the form field tag for PassPhrase would be `passphrase.jsp`.

Rules: You can write your own JSP files to process form fields, but they must conform to the following rules:

- The HTML form field, whether it is a select field, an input field, a text area field, or a hidden form field, must have the same name as the form field tag, but in lowercase. For example, for the tag

  ```
  <tns:CommonName formtype="AdminSpecified" optional="true"
  JSPfilename=”cn.jsp”/>
  ```

  the form field must have the name `commonname` in the file `cn.jsp`.
- The HTML form field must contain a JavaScript function with the name `Valid` concatenated with the lowercase form field name. For example, for the tag

  ```
  <tns:CommonName formtype="AdminSpecified" optional="true"
  JSPfilename=”cn.jsp”/>
  ```

  `cn.jsp` must contain a JavaScript method named `Validcommonname`. This JavaScript method should verify the form field and return `true` if it is valid and `false` if not. If there is no verification to be done, the Valid*formfieldname* method can return true in all cases.

Each of the attributes (`formtype`, `initvalue`, `optional` and `JSPfilename`) can be omitted and defaulted.

Examples of form field tags:

```
<tns:CommonName formtype="AdminSpecified" optional="true" JSPfilename="cn.jsp"/>
<tns:PassPhrase />
```

Roadmap for implementing the PKI Services web application using JSPs in Traditional WebSphere Application Server

About this task

To implement the PKI Services web application using JSPs, perform the tasks in Table 44 on page 242.

Table 44. Task roadmap for implementing the PKI Services web application using the JSPs in Traditional WebSphere Application Server

Subtask	Associated instructions (see . . .)
Preparation	"Steps for preparing to implement the PKI Services web application using JSPs in Traditional WebSphere Application Server" on page 242
Give WebSphere users authorization to use PKI Services functions	"Giving WebSphere users authorization to use PKI Services functions" on page 243
Set up a WebSphere SSL configuration that uses client authentication, to allow WebSphere users to renew and revoke browser certificates	"Allowing WebSphere users to renew and revoke browser certificates" on page 247
Customize the web application	"Customizing the PKI Services web application" on page 257
• Update the template file.	• "Updating the template file" on page 258
• (Optional) Modify the JSP files and update the EAR file with the modified JSP files.	• "(Optional) Modifying the JSP files and the EAR file" on page 258
• Deploy the EAR file to a WebSphere Application Server.	• "Deploying the EAR file to a WebSphere Application Server" on page 261

Steps for preparing to implement the PKI Services web application using JSPs in Traditional WebSphere Application Server

About this task

Perform the following steps to prepare to implement the PKI Services web application using JSPs.

Procedure

1. Set the _PKISERV_ENABLE_JSP environment variable to indicate that you are using the JSPs instead of the REXX CGIs. To do this, uncomment the following line in the environment variables file `pkiserv.envars`:

   ```
   _PKISERV_ENABLE_JSP=TRUE
   ```

   (If the line is not in your copy of the environment variables file, add it.)
2. If you have not previously done so, copy the XML template file and the XML schema file from the directory in which the MVS programmer installed PKI Services to the runtime directory by entering the following commands from the UNIX command line. The default directories are `/usr/lpp/pkiserv/` and `/etc/pkiserv` respectively. The user ID you use for copying files must have superuser authority.

```
cp -p /install-dir/samples/pkitmpl.xml runtime-dir
cp -p /install-dir/samples/PKIServ.xsd runtime-dir
```

Results

Note: You do not copy the file `pkixgen.tmpl`. PKI Services does not ship a copy of this file in the `samples` directory. You generate `pkixgen.tmpl` from `pkitmpl.xml` after you customize `pkitmpl.xml`.

When you are done, you are ready to customize the PKI Services web application.

Giving WebSphere users authorization to use PKI Services functions

You need to give each WebSphere user that uses the PKI Services web application authorization to use PKI Services functions.

Steps for giving WebSphere users authorization to use PKI Services functions using Traditional WebSphere Application Server

Before you begin

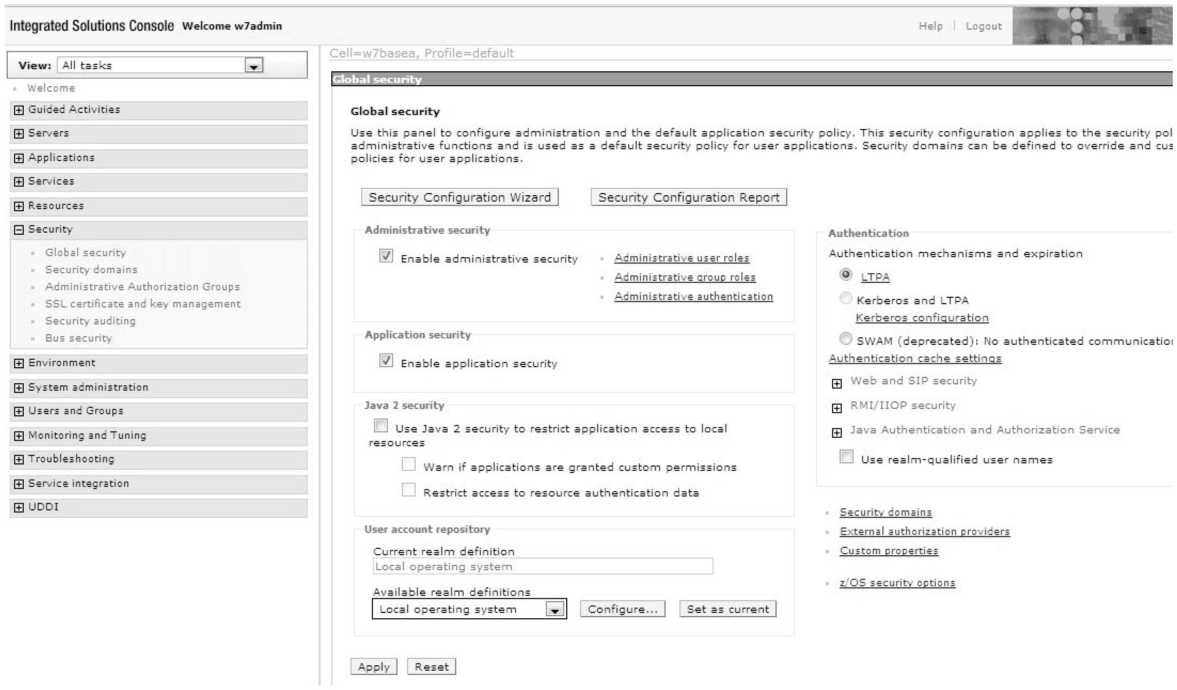
You need RACF administration skills and you must have the RACF SPECIAL attribute. You need WebSphere administration skills.

About this task

Perform the following steps to give WebSphere users authorization to use PKI Services functions using Traditional WebSphere Application Server.

Procedure

1. Log on to the WebSphere administrative console.
2. Configure WebSphere for application security using SAF authorization.
   a) In the left pane, under Security, click **Global security**, and verify that the **Enable application security** check box is selected, as shown in Figure 5 on page 244. If it is not, select it and click **Apply**.


Integrated Solutions Console Welcome w7admin

Help | Logout

View: All tasks

- Welcome
- Guided Activities
- Servers
- Applications
- Services
- Resources
- Security
  - Global security
  - Security domains
  - Administrative Authorization Groups
  - SSL certificate and key management
  - Security auditing
  - Bus security
- Environment
- System administration
- Users and Groups
- Monitoring and Tuning
- Troubleshooting
- Service integration
- UDDI

Cell=w7basea, Profile=default

Global security

Global security

Use this panel to configure administration and the default application security policy. This security configuration applies to the security pol administrative functions and is used as a default security policy for user applications. Security domains can be defined to override and cus policies for user applications.

Security Configuration Wizard | Security Configuration Report

Administrative security

[✓] Enable administrative security

- Administrative user roles
- Administrative group roles
- Administrative authentication

Application security

[✓] Enable application security

Java 2 security

[ ] Use Java 2 security to restrict application access to local resources

[ ] Warn if applications are granted custom permissions

[ ] Restrict access to resource authentication data

User account repository

Current realm definition: Local operating system

Available realm definitions: Local operating system | Configure... | Set as current

Authentication

Authentication mechanisms and expiration

● LTPA

○ Kerberos and LTPA

Kerberos configuration

○ SWAM (deprecated): No authenticated communicatio

Authentication cache settings

- Web and SIP security
- RMI/IIOP security
- Java Authentication and Authorization Service

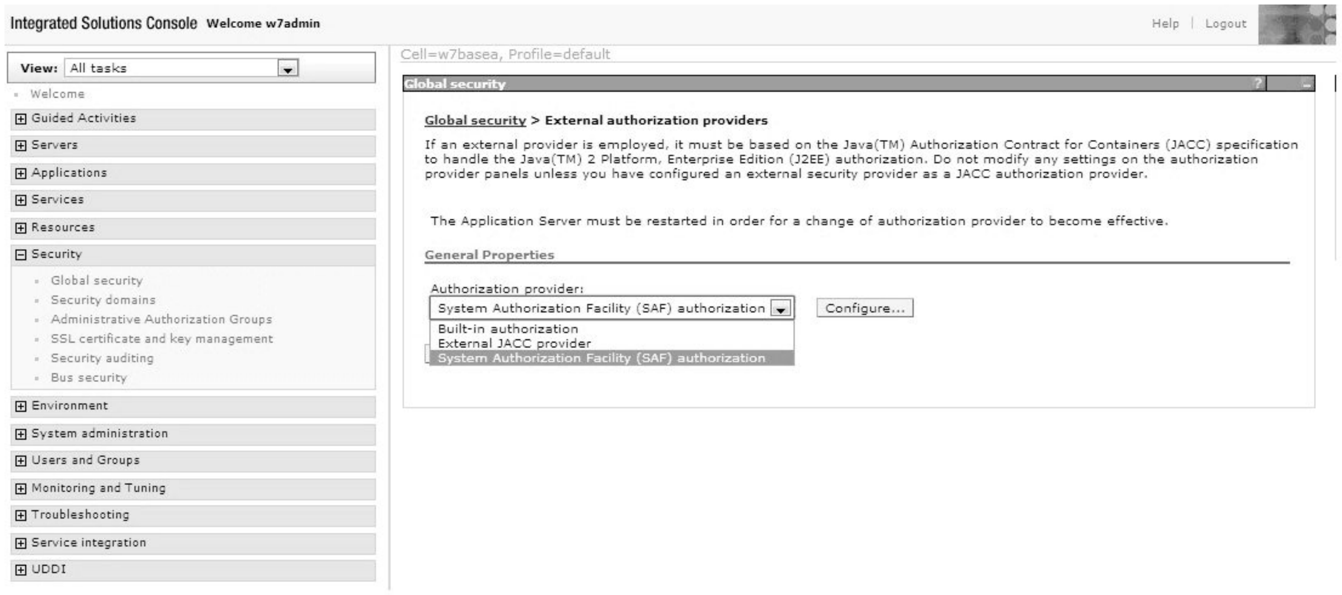
[ ] Use realm-qualified user names

- Security domains
- External authorization providers
- Custom properties
- z/OS security options

Apply | Reset

Figure 5. Configuring WebSphere for application security

b) Click **External authorization providers**, and verify that **System Authorization Facility (SAF) authorization** is selected, as shown in Figure 6 on page 244. If it is not, click **System Authorization Facility (SAF) authorization**, and then click **Configure**. Fill in the **PKI Surrogate User ID** in the **Unauthenticated user ID** field as shown in Figure 7 on page 245.


Integrated Solutions Console Welcome w7admin

Help | Logout

View: All tasks

- Welcome
- Guided Activities
- Servers
- Applications
- Services
- Resources
- Security
  - Global security
  - Security domains
  - Administrative Authorization Groups
  - SSL certificate and key management
  - Security auditing
  - Bus security
- Environment
- System administration
- Users and Groups
- Monitoring and Tuning
- Troubleshooting
- Service integration
- UDDI

Cell=w7basea, Profile=default

Global security

Global security > External authorization providers

If an external provider is employed, it must be based on the Java(TM) Authorization Contract for Containers (JACC) specification to handle the Java(TM) 2 Platform, Enterprise Edition (J2EE) authorization. Do not modify any settings on the authorization provider panels unless you have configured an external security provider as a JACC authorization provider.

The Application Server must be restarted in order for a change of authorization provider to become effective.

General Properties

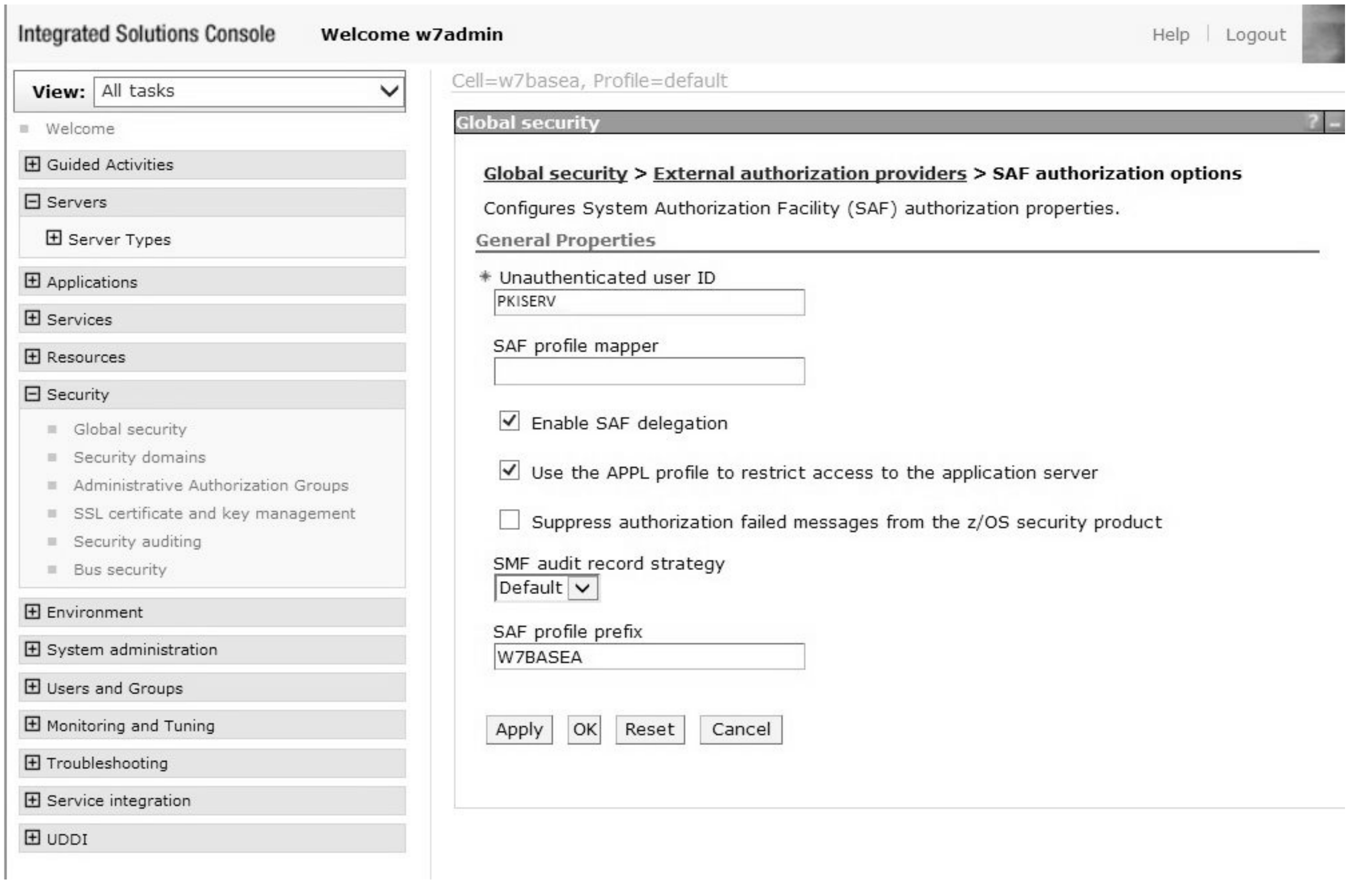
Authorization provider: System Authorization Facility (SAF) authorization | Configure...

- Built-in authorization
- External JACC provider
- System Authorization Facility (SAF) authorization

Figure 6. Configuring Websphere for application security using SAF authorization

Click **Apply** and then **Save** to go back to the page as displayed in Figure 6 on page 244.

Click **Apply** and then **Save** to save the changes to the master configuration.



Cell=w7basea, Profile=default

Global security

Global security > External authorization providers > SAF authorization options

Configures System Authorization Facility (SAF) authorization properties.

General Properties

* Unauthenticated user ID: PKISERV

SAF profile mapper:

[✓] Enable SAF delegation

[✓] Use the APPL profile to restrict access to the application server

[ ] Suppress authorization failed messages from the z/OS security product

SMF audit record strategy: Default

SAF profile prefix: W7BASEA

Figure 7. SAF authorization options

c) From the web page shown in Figure 5 on page 244, click **z/OS security options** and select the **Enable application server and z/OS thread identity synchronization** check box, then click **OK**.

d) If you changed any settings, you see the message shown in Figure 25 on page 263 indicating that changes are made to your local configuration. Click **Save** to save the changes to the master configuration.

3. The `web.xml` file for PKIServ_Web defines three roles: SAFuser, PKIAdmin and PKISurrogate. You need the SAFuser role only if you want to specify z/OS authentication (either zAuthRunAsSurrogate or zAuthRunAsClient) for certificate request or retrieval authentication. In the sample `pkitmpl.xml` file, z/OS authentication is used for these certificate templates: 1-Year SAF Browser Certificate, 1-Year SAF Server Certificate, 2-Year PKI Browser Certificate For Authenticating To z/OS, 5-Year SCEP Certificate - Preregistration, 2-Year EST Certificate - Preregistration and 5-Year PKI Intermediate CA Certificate. The PKIAdmin role allows users assigned it to use the PKI Services administration web pages.

   Enter the following TSO commands to create EJBROLE profiles to map the SAFuser and PKIAdmin roles. The first RDEFINE command creates the SAFuser role and gives it to all users who are authenticated. The second RDEFINE command creates the PKIAdmin role.

```
SETROPTS CLASSACT(EJBROLE)
RDEFINE EJBROLE SAFuser UACC(READ)
RDEFINE EJBROLE PKIAdmin UACC(NONE)
```

   Then enter *one* of the following commands. The first PERMIT command must be entered for *each* user who should have authorization to use the PKI Services administration web pages.

```
PERMIT PKIAdmin CLASS(EJBROLE) ID(userid) ACCESS(READ)
```

or

```
PERMIT PKIAdmin CLASS(EJBROLE) ID(pkigrp) ACCESS(READ)
```

where *pkigrp* is the value of the variable `pkigroup` in the IKYSETUP exec (default value PKIGRP). For a description of the `pkigroup` variable, see Table 19 on page 55.

Create a EJBROLE profile to map the PKI Services surrogate role and specify user ID as APPLDATA. Change the value PKISERV to the user ID you specified for the surrogate user in the IKYSETUP exec, if you specified a different user ID (See Table 19 on page 55).

```
RDEFINE EJBROLE PKISurrogate UACC(READ) APPLDATA('PKISER')
```

Refresh the EJBROLE class after all of the profile updates:

```
SETROPTS RACLIST(EJBROLE) REFRESH
```

Note: If your installation uses security domains the role name is qualified by the security domain, for example:

```
RDEFINE EJBROLE securitydomain.SAFuser UACC(READ)
RDEFINE EJBROLE securitydomain.PKIAdmin UACC(NONE)
PERMIT securitydomain.PKIAdmin CLASS(EJBROLE) ID(PKIGRP) ACCESS(READ)
```

4. Issue TSO commands to give WebSphere users authorization to use PKI Services.
   a) Give the user ID for the WebSphere servant region address space READ access to the FACILITY class profile IRR.RPKISERV.*function.ca-domain*. ASSR1 is the user ID for the WebSphere servant region address space. For unnamed domains issue the command:

   ```
   PERMIT IRR.RPKISERV.* CLASS(FACILITY) ID(ASSR1) ACCESS(READ)
   ```

   For named domains issue the command:

   ```
   PERMIT IRR.RPKISERV.*.DomainName CLASS(FACILITY) ID(ASSR1) ACCESS(READ)
   ```

   b) Authorize the PKI Services surrogate user to use the R_PKIServ functions REQCERT, GENCERT, ADD, VERIFY, REVOKE, RESPOND, SCEPREQ, REQRENEW, GENRENEW, EXPORT, and QRECOVER. Enter the following TSO commands. Change the value PKISERV to the user ID you specified for the surrogate user in the IKYSETUP exec, if you specified a different user ID. (See Table 19 on page 55.)

   ```
   RDEFINE FACILITY IRR.DIGTCERT.REQCERT UACC(NONE)
   PERMIT IRR.DIGTCERT.REQCERT CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

   RDEFINE FACILITY IRR.DIGTCERT.GENCERT UACC(NONE)
   PERMIT IRR.DIGTCERT.GENCERT CLASS(FACILITY) ID(PKISERV) ACCESS(CONTROL)

   RDEFINE FACILITY IRR.DIGTCERT.ADD UACC(NONE)
   PERMIT IRR.DIGTCERT.ADD CLASS(FACILITY) ID(PKISERV) ACCESS(UPDATE)

   RDEFINE FACILITY IRR.DIGTCERT.VERIFY UACC(NONE)
   PERMIT IRR.DIGTCERT.VERIFY CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

   RDEFINE FACILITY IRR.DIGTCERT.REVOKE UACC(NONE)
   PERMIT IRR.DIGTCERT.REVOKE CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

   RDEFINE FACILITY IRR.DIGTCERT.RESPOND UACC(NONE)
   PERMIT IRR.DIGTCERT.RESPOND CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

   RDEFINE FACILITY IRR.DIGTCERT.SCEPREQ UACC(NONE)
   PERMIT IRR.DIGTCERT.SCEPREQ CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

   RDEFINE FACILITY IRR.DIGTCERT.REQRENEW UACC(NONE)
   PERMIT IRR.DIGTCERT.REQRENEW CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

   RDEFINE FACILITY IRR.DIGTCERT.GENRENEW UACC(NONE)
   PERMIT IRR.DIGTCERT.GENRENEW CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

   RDEFINE FACILITY IRR.DIGTCERT.EXPORT UACC(NONE)
   PERMIT IRR.DIGTCERT.EXPORT CLASS(FACILITY) ID(PKISERV) ACCESS(CONTROL)
   ```

```
RDEFINE FACILITY IRR.DIGTCERT.QRECOVER UACC(NONE)
PERMIT IRR.DIGTCERT.QRECOVER CLASS(FACILITY) ID(PKISERV) ACCESS(READ)
```

c) If you modified `pkitmpl.xml` and use the zAuthRunAsClient retrieve authentication type:

```
<tns:retrieve_authtype>zAuthRunAsClient</tns:retrieve_authtype>
```

you must give any user ID that requests this kind of certificate authorization to the R_PKIServ EXPORT function:

```
PERMIT IRR.DIGTCERT.EXPORT CLASS(FACILITY) ID(userid) ACCESS(CONTROL)
```

d) Allow the WebSphere user ID to run as an authenticated RACF user. To do this, the user ID for the WebSphere control region must have CONTROL access to the resource BBO.SYNC.*cell_short_name.cluster_short_name*. Enter commands similar to the following example:

```
RDEFINE FACILITY BBO.SYNC.DCEIMGLX.BBOC001 UACC(NONE)
PERMIT BBO.SYNC.DCEIMGLX.BBOC001 CLASS(FACILITY) ID(ASCR1) ACCESS(CONTROL)
```

e) Refresh the in-storage profiles so that the changes you made to the FACILITY class take effect:

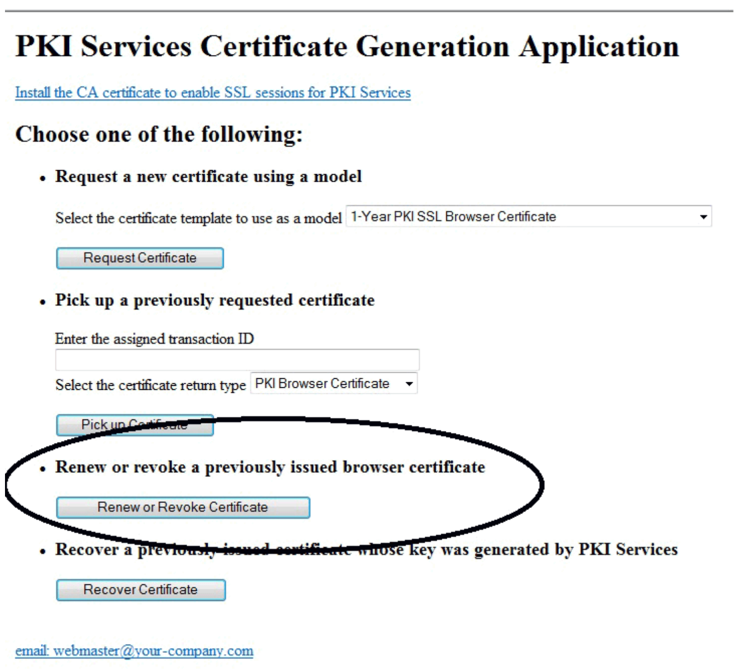
```
SETR RACLIST(FACILITY) REFRESH
```

Results

When you are done, you authorized WebSphere users to use PKI Services functions.

Allowing WebSphere users to renew and revoke browser certificates

WebSphere sets up server authentication automatically with the SSL certificate and key ring that is created during installation. To allow WebSphere users to renew and revoke browser certificates (see Figure 8 on page 248), you need to set up a WebSphere SSL configuration that uses client authentication. You may use the same SSL certificate and key ring that is used for server authentication for client authentication or you can create a new SSL certificate and key ring.


PKI Services Certificate Generation Application

Install the CA certificate to enable SSL sessions for PKI Services

Choose one of the following:

- Request a new certificate using a model

  Select the certificate template to use as a model: 1-Year PKI SSL Browser Certificate

  Request Certificate

- Pick up a previously requested certificate

  Enter the assigned transaction ID

  Select the certificate return type: PKI Browser Certificate

  Pick up Certificate

- Renew or revoke a previously issued browser certificate

  Renew or Revoke Certificate

- Recover a previously issued certificate whose key was generated by PKI Services

  Recover Certificate

email: webmaster@your-company.com

Figure 8. Renewing or revoking a browser certificate

Steps for allowing WebSphere users to renew and revoke browser certificates

Perform the following steps to set up a WebSphere SSL configuration that uses client authentication, so that users can renew and revoke browser certificates.

Before you begin

You need RACF administration skills. You must have the RACF SPECIAL attribute or sufficient authority to IRR.DIGTCERT resources in the FACILITY class to issue the RACDCERT commands shown.

If you want to create a new SSL certificate and a new key ring, follow Step 1 and 2.

Procedure

1. **(Optional)** Create a RACF key ring that contains the PKI Services CA certificate for the WebSphere Control address space user ID (ASCR1 in this example):

```
RACDCERT ADDRING(WASKeyring.PKI) ID(ASCR1)
```

2. **(Optional)** Create a server certificate that is signed by the PKI CA certificate, and add this certificate to the key ring.

```
RACDCERT GENCERT ID(ASCR1)
SUBJECTSDN(CN(<www.YourCompany.com>) O(<Your Company>) L(<Your City>)
SP(<Your Full State or Province Name>) C(<Your Country 2 Letter
Abbreviation>))
NOTAFTER(DATE(2020/12/31))
WITHLABEL('pki ssl cert')
SIGNWITH(CERTAUTH LABEL('Local PKI CA')) TRUST
RACDCERT ID(ASCR1)
CONNECT(ID(ASCR1) LABEL('pki ssl cert') RING(WASKeyring.PKI)
USAGE(PERSONAL))
```

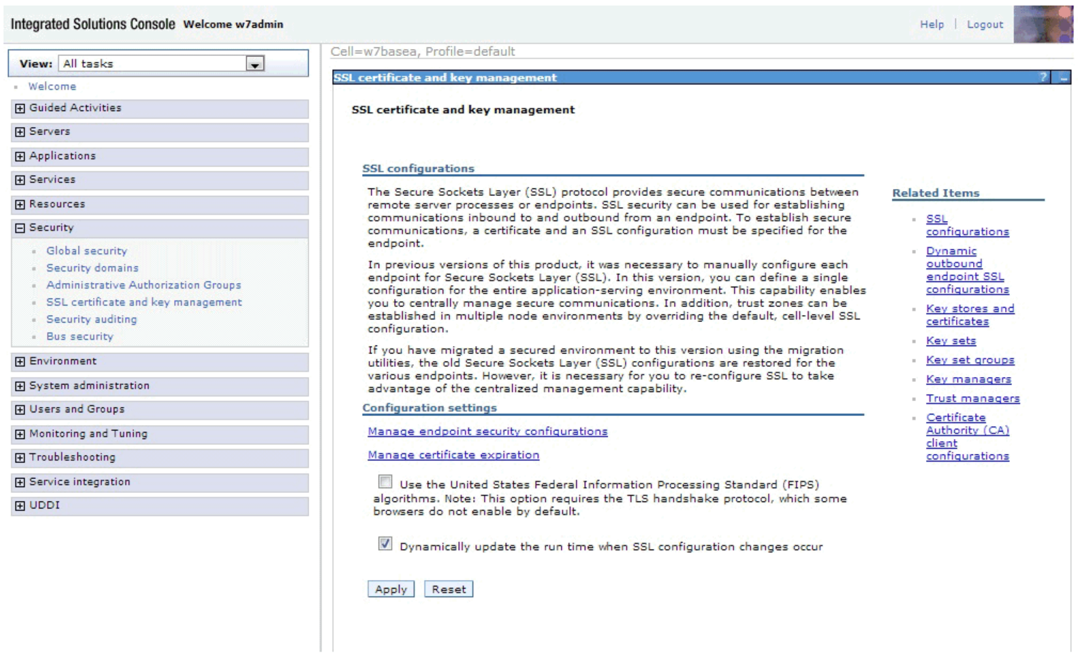
3. If you created a new key ring in Step 1, use the following:

   ```
   RACDCERT ID(ASCR1) CONNECT(CERTAUTH LABEL('Local PKI CA')
   RING(WASKeyring.PKI))
   ```

   If you use the key ring that was created during WebSphere installation, use the following:

   ```
   RACDCERT ID(ASCR1) CONNECT(CERTAUTH LABEL('Local PKI CA')
   RING(<WAS control region key ring>))
   ```

4. Define the RACF keystore to WebSphere. On the WebSphere administrator console, on the left side of the page expand **Security** and click **SSL certificate and key management**. You should see a web page that looks like Figure 9 on page 249.



SSL certificate and key management

SSL configurations

The Secure Sockets Layer (SSL) protocol provides secure communications between remote server processes or endpoints. SSL security can be used for establishing communications inbound to and outbound from an endpoint. To establish secure communications, a certificate and an SSL configuration must be specified for the endpoint.

In previous versions of this product, it was necessary to manually configure each endpoint for Secure Sockets Layer (SSL). In this version, you can define a single configuration for the entire application-serving environment. This capability enables you to centrally manage secure communications. In addition, trust zones can be established in multiple node environments by overriding the default, cell-level SSL configuration.

If you have migrated a secured environment to this version using the migration utilities, the old Secure Sockets Layer (SSL) configurations are restored for the various endpoints. However, it is necessary for you to re-configure SSL to take advantage of the centralized management capability.

Configuration settings

Manage endpoint security configurations

Manage certificate expiration

☐ Use the United States Federal Information Processing Standard (FIPS) algorithms. Note: This option requires the TLS handshake protocol, which some browsers do not enable by default.

☑ Dynamically update the run time when SSL configuration changes occur

Apply
Reset
Related Items
SSL configurations
Dynamic outbound endpoint SSL configurations
Key stores and certificates
Key sets
Key set groups
Key managers
Trust managers
Certificate Authority (CA) client configurations

Figure 9. WebSphere SSL certificate and key management page

   Click **Key stores and certificates** on the right side of the page. On the next page click **New** to create a new keystore. On the next page, enter a name for your keystore in the **Name** field. In the **Path** field enter:

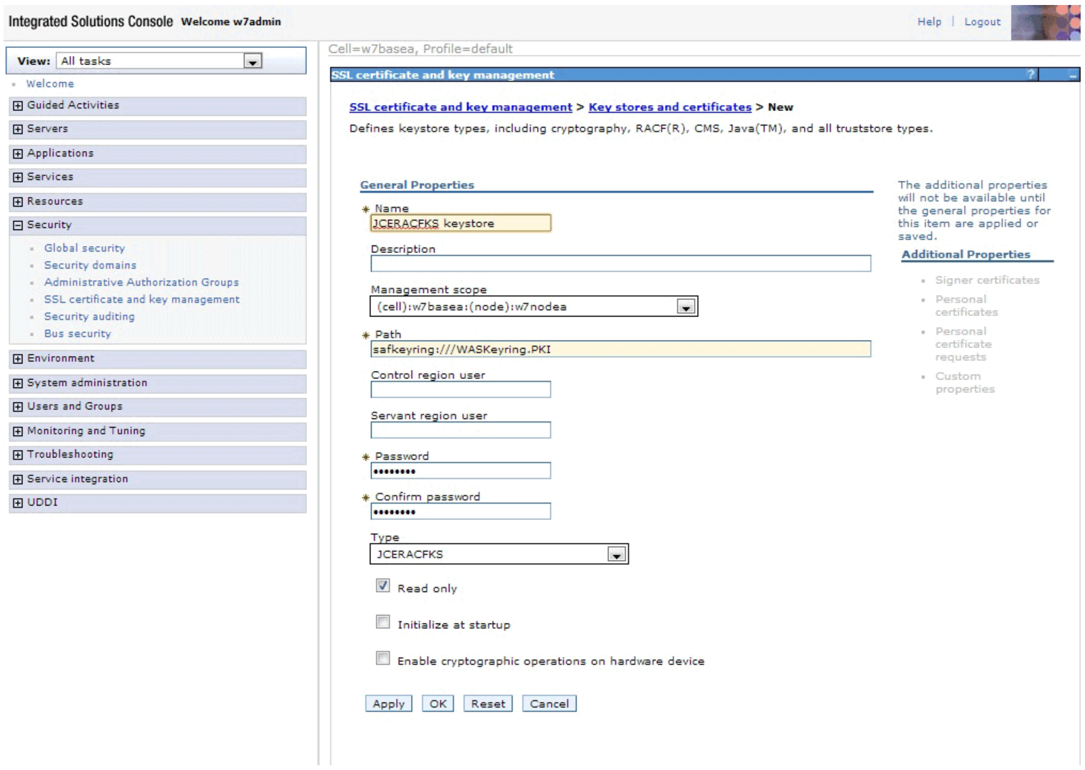
   ```
   safkeyring:///WASKeyring.PKI
   ```

   if you create a new ring, or

   ```
   safkeyring:///<WAS control region key ring>
   ```

   if you use the key ring created during WebSphere installation.

   In the **Password** and **Confirm password** fields enter a password value of `password`. Select the **Read only** check box. See Figure 10 on page 250


Integrated Solutions Console Welcome w7admin

Cell=w7basea, Profile=default

SSL certificate and key management

SSL certificate and key management > Key stores and certificates > New

Defines keystore types, including cryptography, RACF(R), CMS, Java(TM), and all truststore types.

General Properties

* Name: JCERACFKS keystore

Description:

Management scope: (cell):w7basea:(node):w7nodea

* Path: safkeyring:///WASKeyring.PKI

Control region user:

Servant region user:

* Password: ••••••••

* Confirm password: ••••••••

Type: JCERACFKS

[✓] Read only

[ ] Initialize at startup

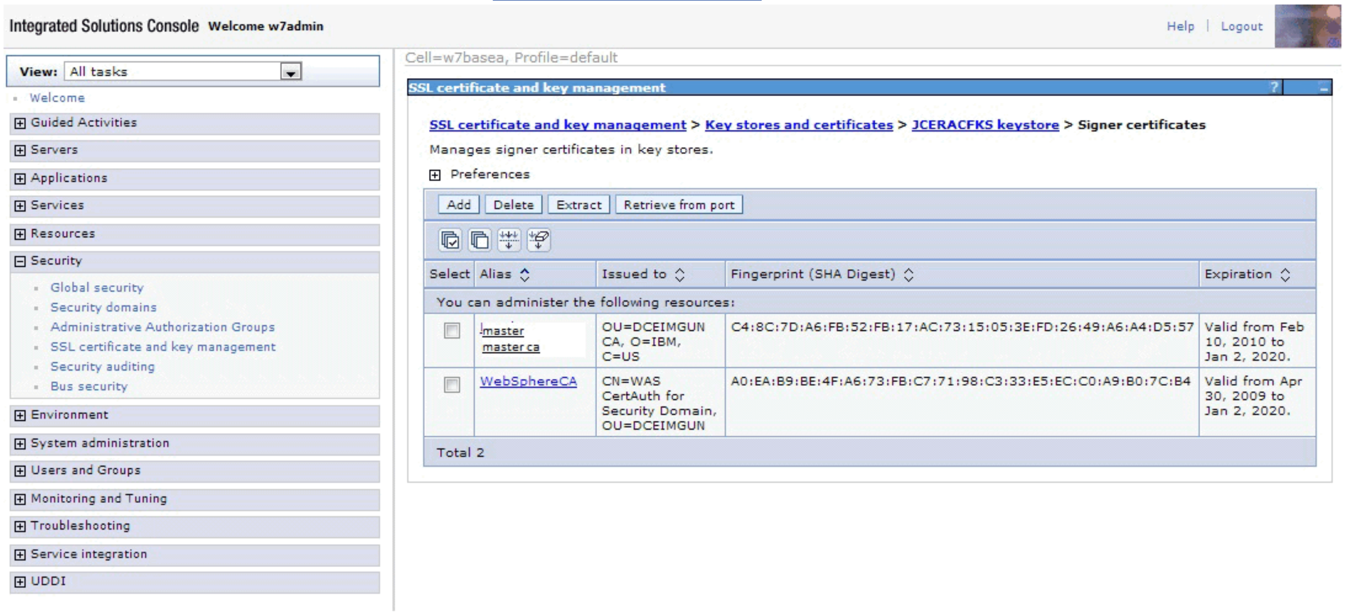
[ ] Enable cryptographic operations on hardware device

The additional properties will not be available until the general properties for this item are applied or saved.

Additional Properties
- Signer certificates
- Personal certificates
- Personal certificate requests
- Custom properties

Figure 10. WebSphere page for creating a new keystore

Click **OK** and then **Save**. You should now be able to click the name of the keystore you created and view the signer certificate. See Figure 11 on page 250.


Integrated Solutions Console Welcome w7admin

Cell=w7basea, Profile=default

SSL certificate and key management

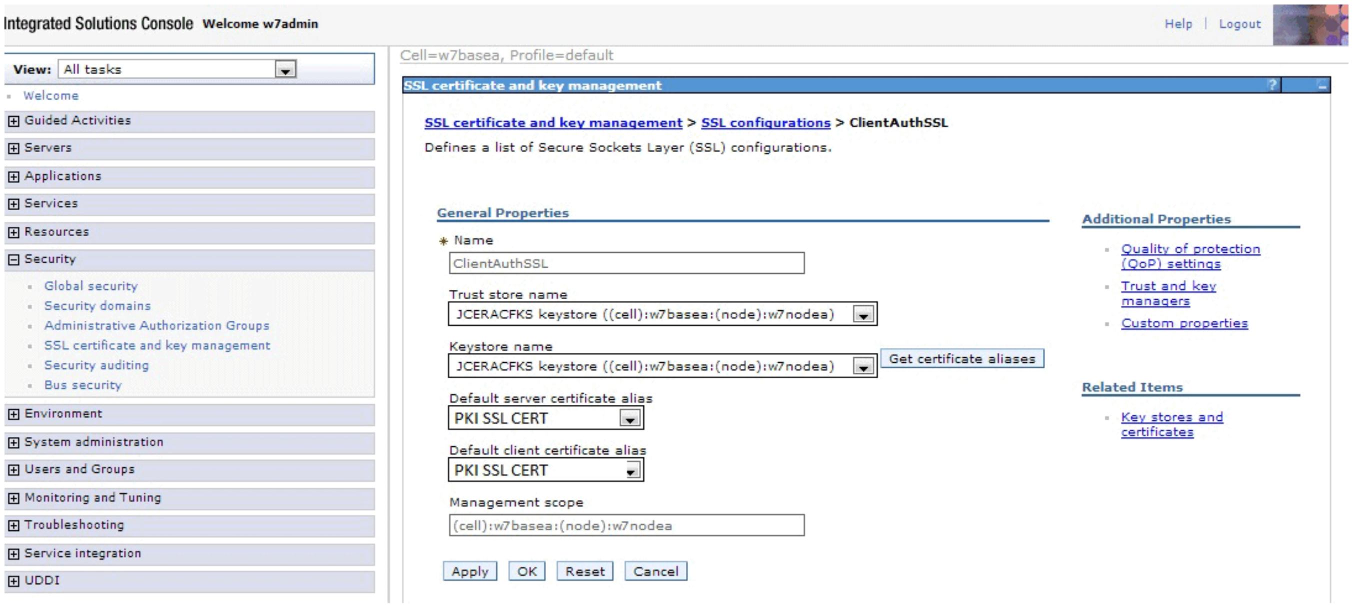
SSL certificate and key management > Key stores and certificates > JCERACFKS keystore > Signer certificates

Manages signer certificates in key stores.

Select	Alias	Issued to	Fingerprint (SHA Digest)	Expiration
You can administer the following resources:				
[ ]	master master ca	OU=DCEIMGUN CA, O=IBM, C=US	C4:8C:7D:A6:FB:52:FB:17:AC:73:15:05:3E:FD:26:49:A6:A4:D5:57	Valid from Feb 10, 2010 to Jan 2, 2020.
[ ]	WebSphereCA	CN=WAS CertAuth for Security Domain, OU=DCEIMGUN	A0:EA:B9:BE:4F:A6:73:FB:C7:71:98:C3:33:E5:EC:C0:A9:B0:7C:B4	Valid from Apr 30, 2009 to Jan 2, 2020.
Total 2				

Figure 11. Viewing the signer certificate

5. Create a new JSSE type SSL configuration. On the left side of the page expand **Security** and click **SSL certificate and key management**. On the right side of the page click **SSL configurations**. (See Figure 9 on page 249.) The page shown in Figure 12 on page 251 is displayed.



Defines a list of Secure Sockets Layer (SSL) configurations.

General Properties

* Name: ClientAuthSSL

Trust store name: JCERACFKS keystore ((cell):w7basea:(node):w7nodea)

Keystore name: JCERACFKS keystore ((cell):w7basea:(node):w7nodea)

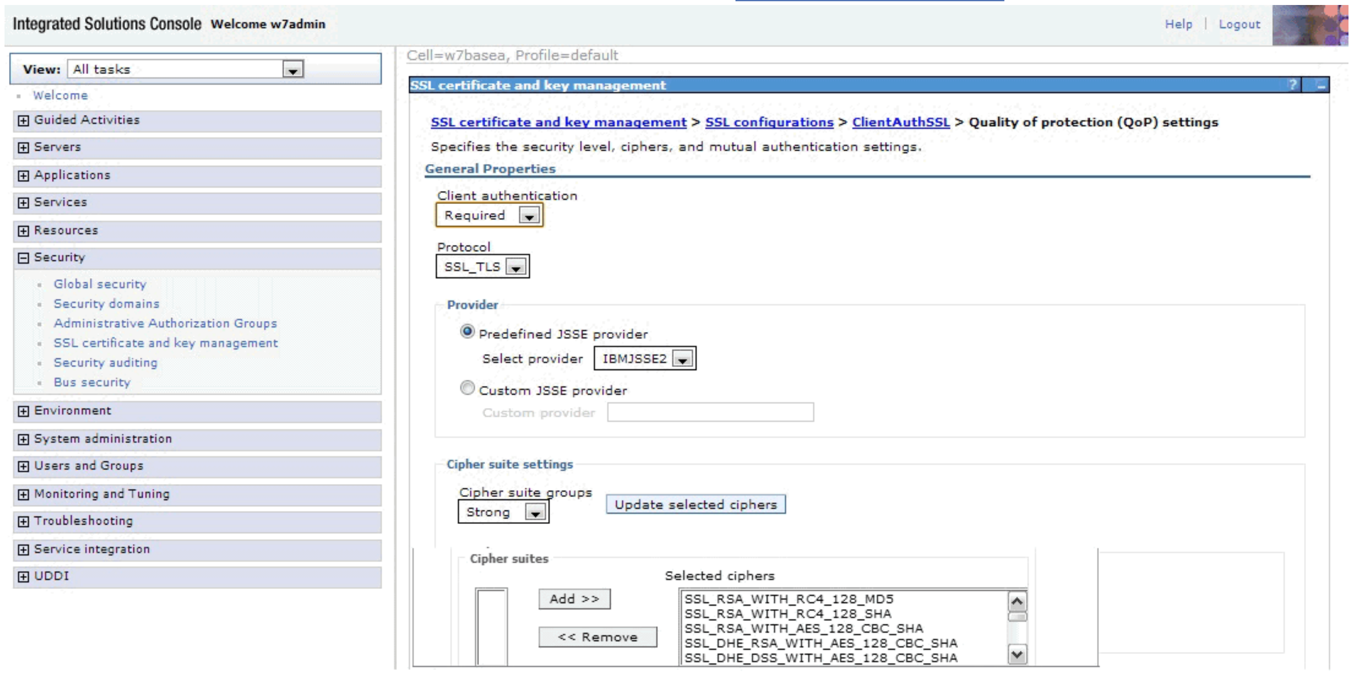
Default server certificate alias: PKI SSL CERT

Default client certificate alias: PKI SSL CERT

Management scope: (cell):w7basea:(node):w7nodea

Figure 12. WebSphere new SSL configuration page

In the **Name** field enter `ClientAuthSSL`. In the **Trust store name** and **Keystore name** fields, enter the name of the keystore that you just defined. Click **Get certificate aliases** to get the alias (label) of the default certificate in your key ring. This is the value for Default server certificate alias and Default client certificate alias. Click **OK**. The page shown in Figure 13 on page 251 is displayed.



Specifies the security level, ciphers, and mutual authentication settings.

General Properties

Client authentication: Required

Protocol: SSL_TLS

Provider

(●) Predefined JSSE provider

Select provider: IBMJSSE2

( ) Custom JSSE provider

Custom provider:

Cipher suite settings

Cipher suite groups: Strong

Cipher suites

Selected ciphers:

SSL_RSA_WITH_RC4_128_MD5
SSL_RSA_WITH_RC4_128_SHA
SSL_RSA_WITH_AES_128_CBC_SHA
SSL_DHE_RSA_WITH_AES_128_CBC_SHA
SSL_DHE_DSS_WITH_AES_128_CBC_SHA

Figure 13. WebSphere quality of protection settings page

In the **Client authentication** list, select **Required**, then click **OK**.

Step 5a:

a. Click on "ClientAuthSSL" SSL configuration.
b. Click on "Quality of Protection (QoP) Settings.
c. For the protocol, ensure that TLSv1.2 is selected.

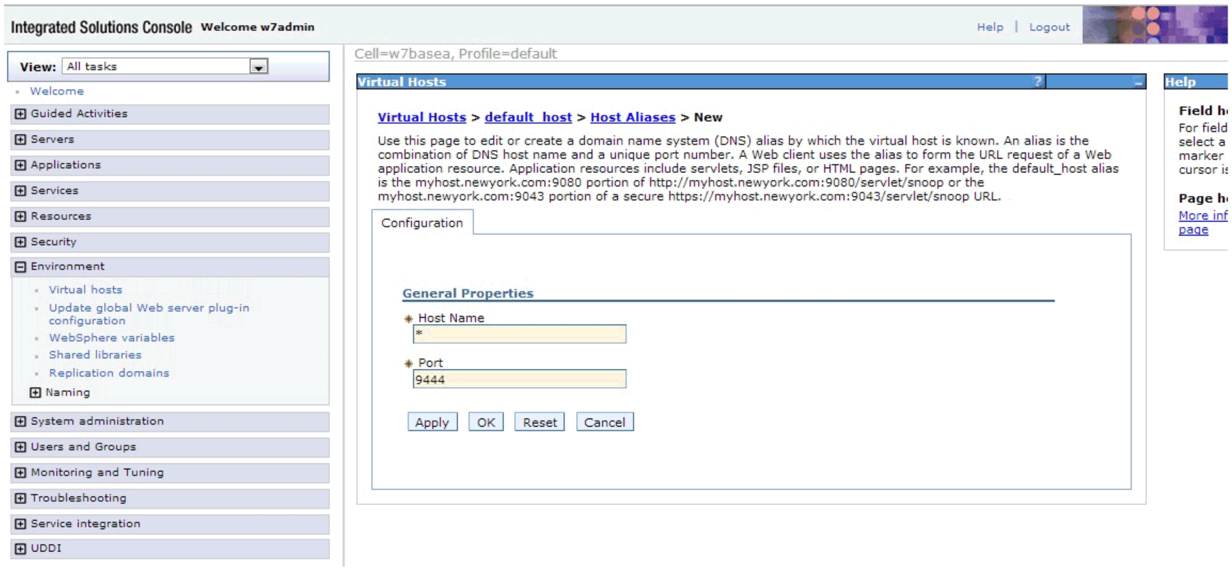
   Note: With the selection of protocol TLS1.2, any certificates using MD2 hashing will be rejected during TLS1.2 handshake.
d. Select "Custom" from the drop down select field for Cipher suite groups.
e. Click on "Update selected ciphers".
f. Ensure that the following ciphers are added to the configuration. To add a cipher, select the cipher from the list and click on "Add".
   1) SSL_RSA_WITH_AES_128_GCM_SHA256
   2) SSL_RSA_WITH_AES_256_GCM_SHA384
   3) SSL_RSA_WITH_AES_128_CBC_SHA256
   4) SSL_RSA_WITH_AES_256_CBC_SHA256
   5) SSL_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
   6) SSL_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
   7) SSL_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
   8) SSL_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
   9) SSL_ECDHE_RSA_WITH_AES_128_CBC_SHA256
   10) SSL_ECDHE_RSA_WITH_AES_256_CBC_SHA384
   11) SSL_ECDHE_RSA_WITH_AES_128_GCM_SHA256
   12) SSL_ECDHE_RSA_WITH_AES_256_GCM_SHA384
g. Click OK and save directly to the master configuration.

6. Define a new port for your default host. On the left side of the page, expand **Environment** and click **Virtual Hosts**. Then click **default host**, then click **Host Aliases**, then click **New**. On the page that results, the value 9444 in the **Port** field should match your client authorization port address in the `web.xml` file. See Figure 14 on page 253.

   If you must change the authorization port address in the `web.xml` file, follow the instructions in "Steps for creating application domains other than Application2" on page 301, except in steps "3" on page 302 and "4" on page 302 you must find the section that looks like:

```
<env-entry>
         <env-entry-name>_PKISERV_CLIENTAUTH_PORT</env-entry-name>
         <env-entry-type>java.lang.String</env-entry-type>
         <env-entry-value>9444</env-entry-value>
</env-entry>
```

   and change the 9444 to the value that you want to use.



Virtual Hosts > default_host > Host Aliases > New

Use this page to edit or create a domain name system (DNS) alias by which the virtual host is known. An alias is the combination of DNS host name and a unique port number. A Web client uses the alias to form the URL request of a Web application resource. Application resources include servlets, JSP files, or HTML pages. For example, the default_host alias is the myhost.newyork.com:9080 portion of http://myhost.newyork.com:9080/servlet/snoop or the myhost.newyork.com:9043 portion of a secure https://myhost.newyork.com:9043/servlet/snoop URL.

Configuration

General Properties

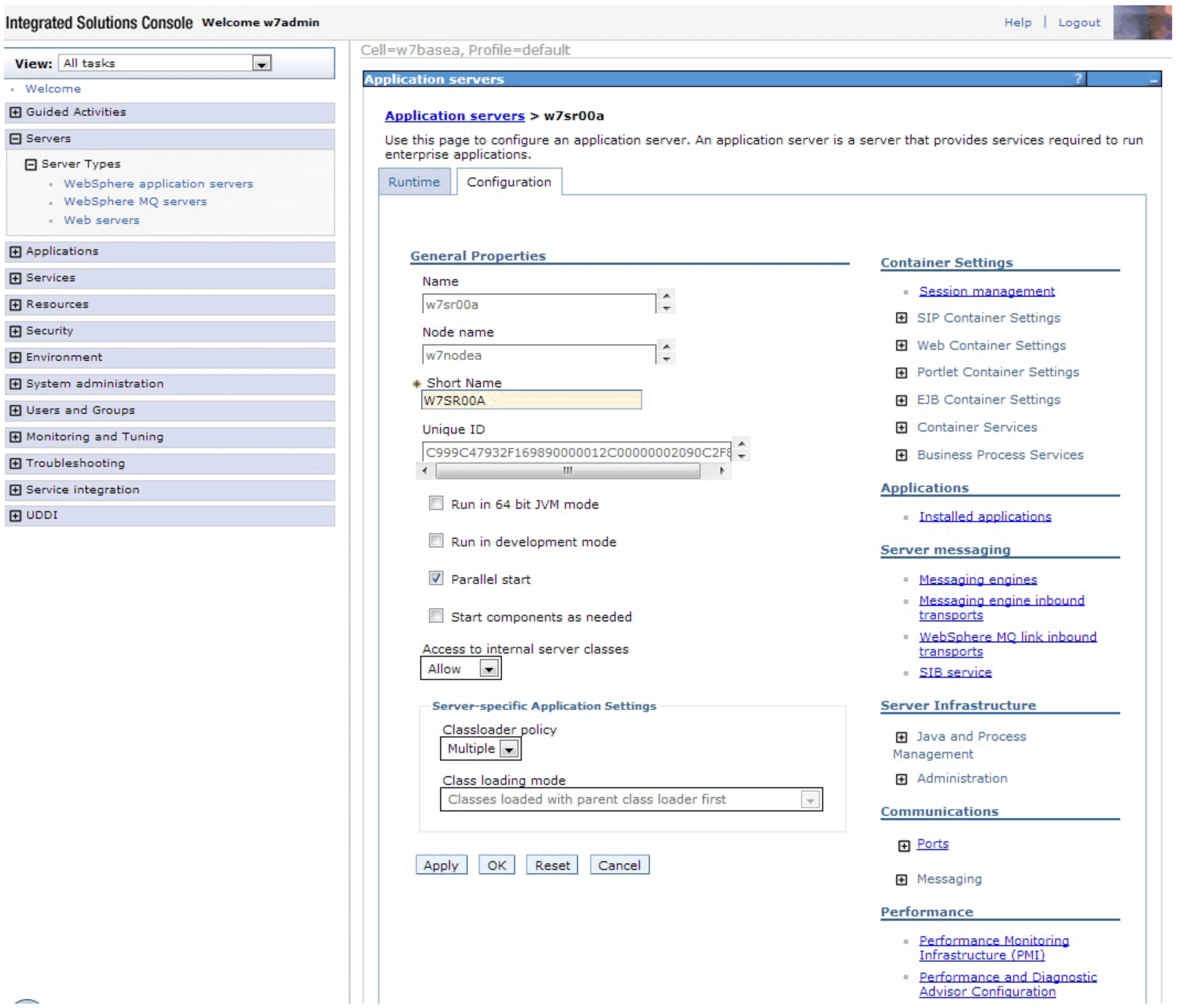
Host Name: *

Port: 9444

Figure 14. Defining a new port

Click **OK** and then **Save** to save this new port number.

7. Assign the new port number to your application server. On the left side of the page, expand **Servers** and click **Websphere application servers**. Click the name of your application server. See Figure 15 on page 254.


Integrated Solutions Console Welcome w7admin

Help | Logout

View: All tasks

- Welcome
- Guided Activities
- Servers
  - Server Types
    - WebSphere application servers
    - WebSphere MQ servers
    - Web servers
- Applications
- Services
- Resources
- Security
- Environment
- System administration
- Users and Groups
- Monitoring and Tuning
- Troubleshooting
- Service integration
- UDDI

Cell=w7basea, Profile=default

Application servers

Application servers > w7sr00a

Use this page to configure an application server. An application server is a server that provides services required to run enterprise applications.

Runtime | Configuration

General Properties

Name: w7sr00a

Node name: w7nodea

* Short Name: W7SR00A

Unique ID: C999C47932F169890000012C00000002090C2F8

- [ ] Run in 64 bit JVM mode
- [ ] Run in development mode
- [✓] Parallel start
- [ ] Start components as needed

Access to internal server classes: Allow

Server-specific Application Settings

Classloader policy: Multiple

Class loading mode: Classes loaded with parent class loader first

Apply | OK | Reset | Cancel

Container Settings
- Session management
- SIP Container Settings
- Web Container Settings
- Portlet Container Settings
- EJB Container Settings
- Container Services
- Business Process Services

Applications
- Installed applications

Server messaging
- Messaging engines
- Messaging engine inbound transports
- WebSphere MQ link inbound transports
- SIB service

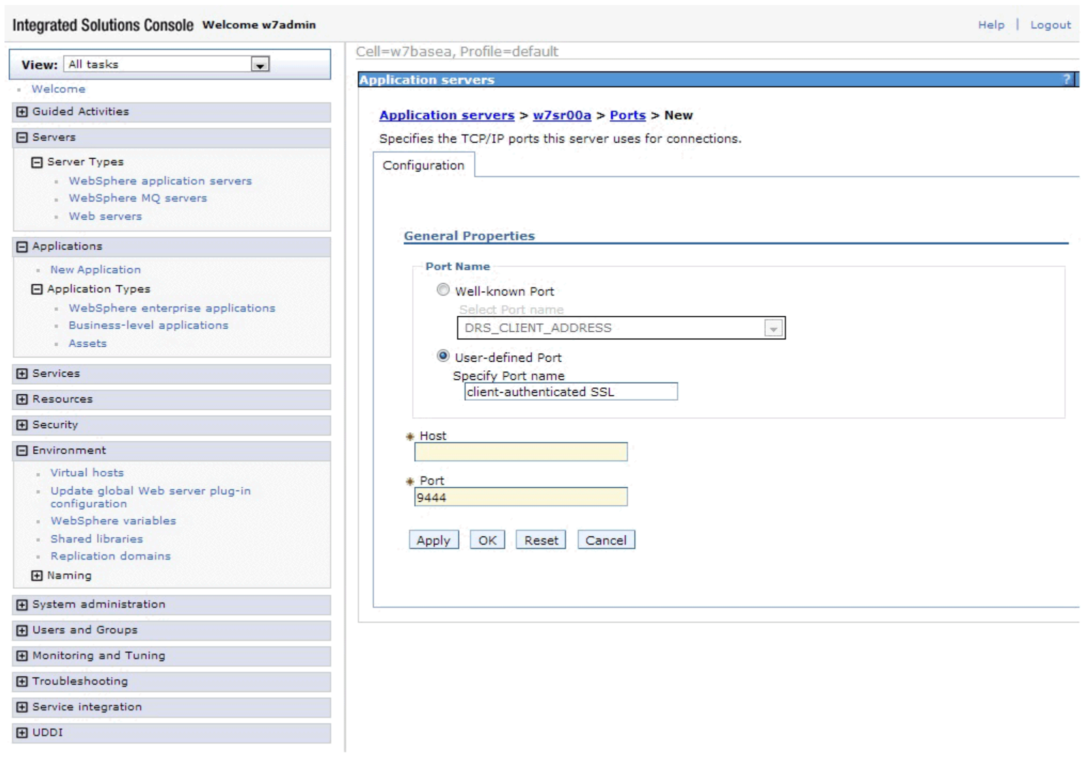
Server Infrastructure
- Java and Process Management
- Administration

Communications
- Ports
- Messaging

Performance
- Performance Monitoring Infrastructure (PMI)
- Performance and Diagnostic Advisor Configuration

Figure 15. Assigning a port to the application server

Under Communications expand **Ports** and click **New**. Click **User-defined Port**, and in the **Specify Port name** field enter `client-authenticated SSL`. In the **Host** field enter `*`. In the **Port** field enter 9444. See Figure 16 on page 255.



Cell=w7basea, Profile=default

Application servers

Application servers > w7sr00a > Ports > New

Specifies the TCP/IP ports this server uses for connections.

Configuration

General Properties

Port Name

Well-known Port

Select Port name: DRS_CLIENT_ADDRESS

User-defined Port

Specify Port name: client-authenticated SSL

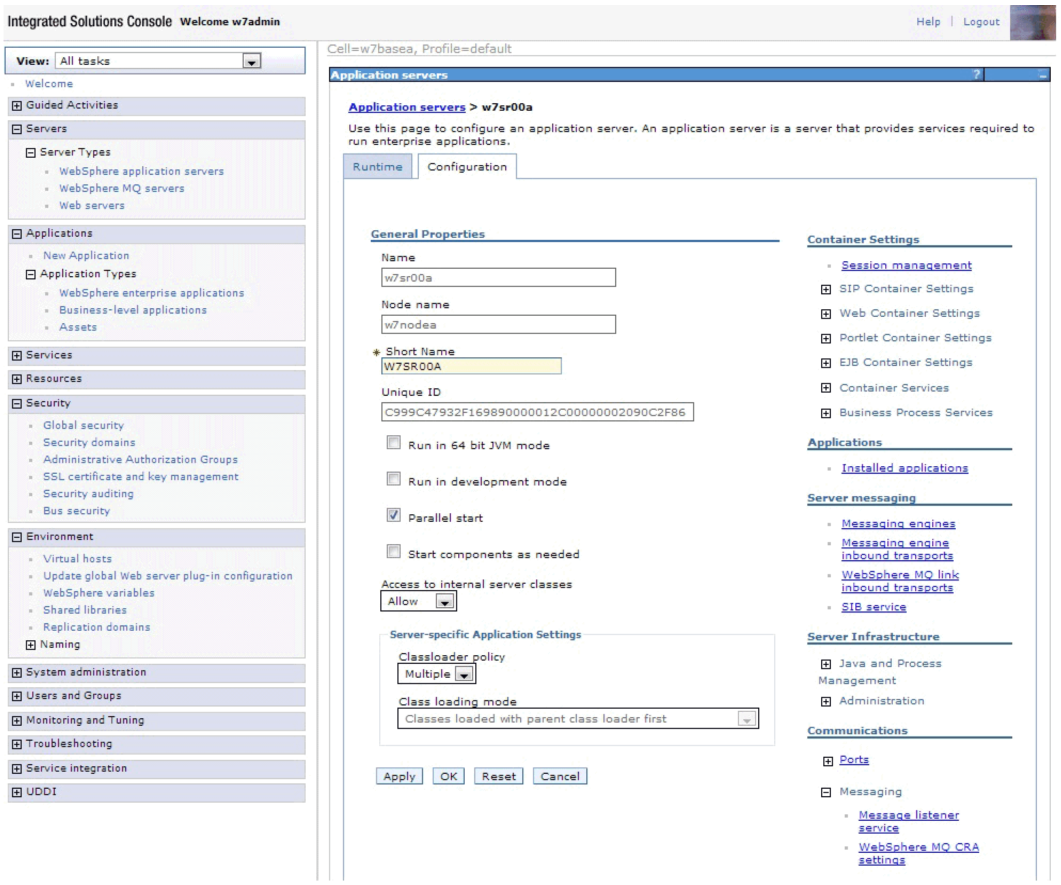
Host:

Port: 9444

Figure 16. Assigning your new port to the application server

Click **OK** and **Save**.

8. Define transport chains associated with the new port.
   a) On the left side of the page expand **Servers** and click **Websphere application servers**. Click the name of your application server. Expand **web container settings**. See Figure 17 on page 256.



Cell=w7basea, Profile=default

Application servers

Application servers > w7sr00a

Use this page to configure an application server. An application server is a server that provides services required to run enterprise applications.

General Properties

Name: w7sr00a

Node name: w7nodea

* Short Name: W7SR00A

Unique ID: C999C47932F169890000012C00000002090C2F86

☐ Run in 64 bit JVM mode

☐ Run in development mode

☑ Parallel start

☐ Start components as needed

Access to internal server classes: Allow

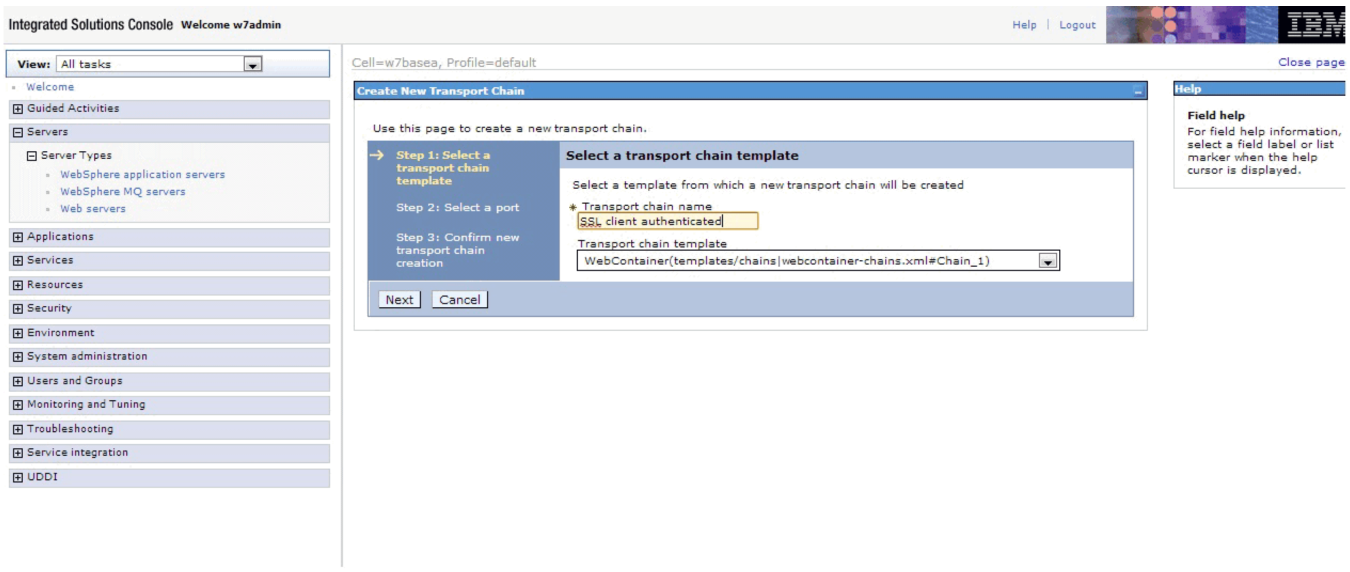
Server-specific Application Settings

Classloader policy: Multiple

Class loading mode: Classes loaded with parent class loader first

Figure 17. Selecting web container transport chains

b) Click **web container transport chains**, and then click **New**. You should see a page like Figure 18 on page 256.



Cell=w7basea, Profile=default

Create New Transport Chain

Use this page to create a new transport chain.

Select a transport chain template

Select a template from which a new transport chain will be created

* Transport chain name: SSL client authenticated

Transport chain template: WebContainer(templates/chains|webcontainer-chains.xml#Chain_1)

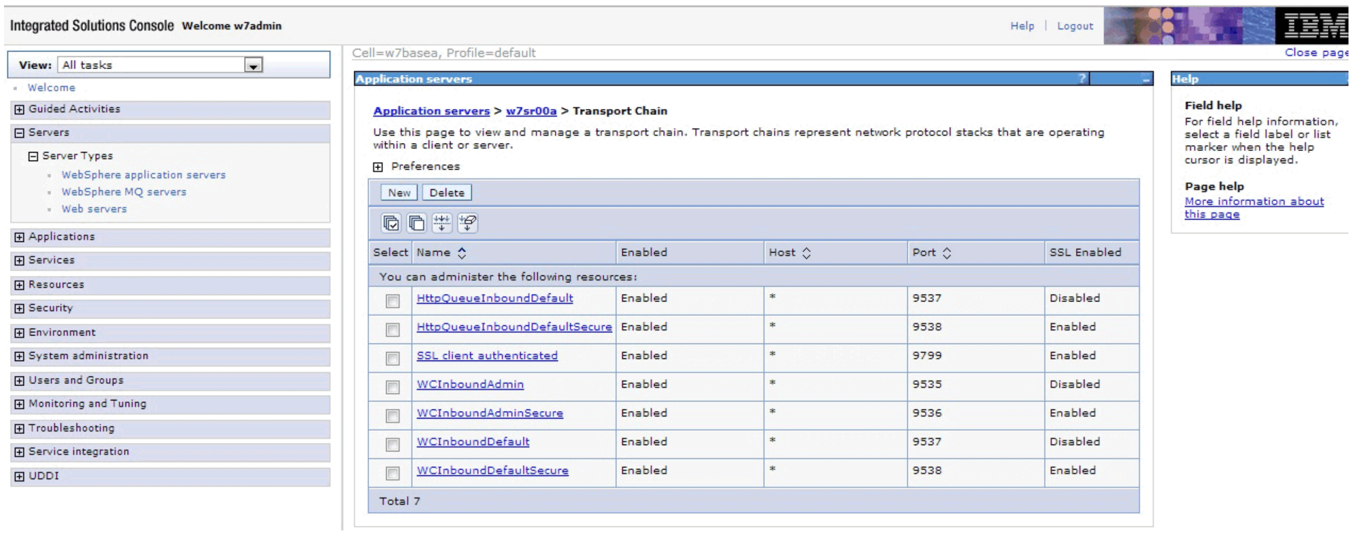
Help

Field help

For field help information, select a field label or list marker when the help cursor is displayed.

Figure 18. Selecting a transport chain template

c) Choose a secure transport chain template and click **Next**. On the next page select **Use an existing port**, then click **Confirm** and then **Save**. A page opens that displays all existing transport chains. See Figure 19 on page 257.


Integrated Solutions Console Welcome w7admin

Help | Logout

View: All tasks

- Welcome
- Guided Activities
- Servers
  - Server Types
    - WebSphere application servers
    - WebSphere MQ servers
    - Web servers
- Applications
- Services
- Resources
- Security
- Environment
- System administration
- Users and Groups
- Monitoring and Tuning
- Troubleshooting
- Service integration
- UDDI

Cell=w7basea, Profile=default

Close page

Application servers

Application servers > w7sr00a > Transport Chain

Use this page to view and manage a transport chain. Transport chains represent network protocol stacks that are operating within a client or server.

Preferences

New Delete

Select	Name	Enabled	Host	Port	SSL Enabled
You can administer the following resources:					
☐	HttpQueueInboundDefault	Enabled	*	9537	Disabled
☐	HttpQueueInboundDefaultSecure	Enabled	*	9538	Enabled
☐	SSL client authenticated	Enabled	*	9799	Enabled
☐	WCInboundAdmin	Enabled	*	9535	Disabled
☐	WCInboundAdminSecure	Enabled	*	9536	Enabled
☐	WCInboundDefault	Enabled	*	9537	Disabled
☐	WCInboundDefaultSecure	Enabled	*	9538	Enabled
Total 7					

Help

Field help

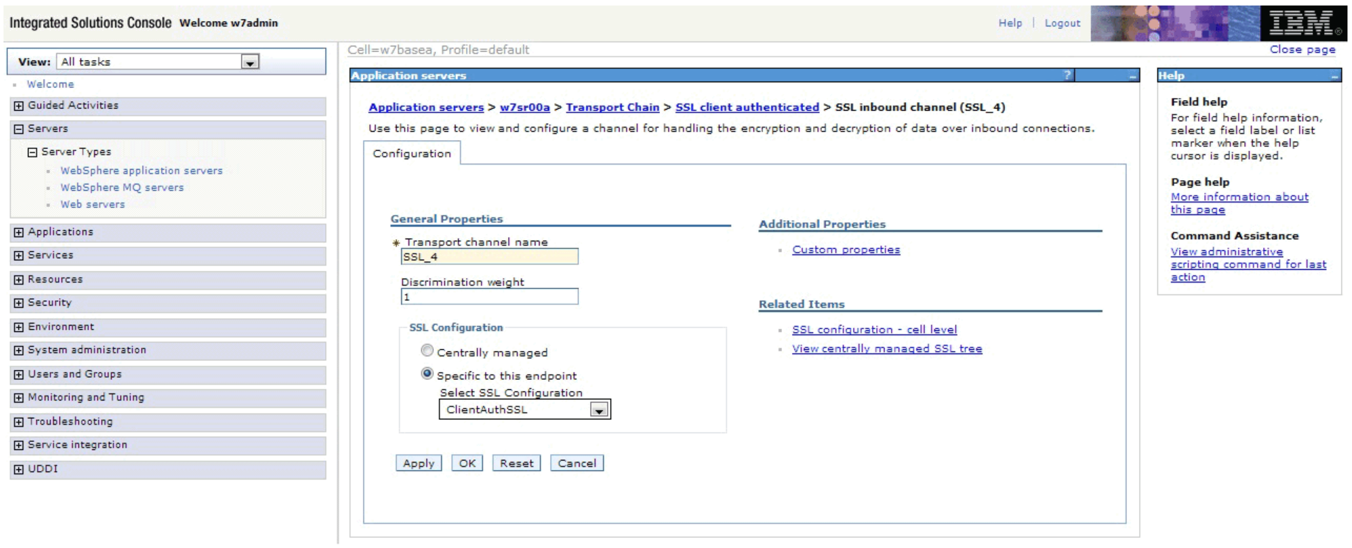
For field help information, select a field label or list marker when the help cursor is displayed.

Page help

More information about this page

Figure 19. All existing transport chains

d) Click the name of the chain you defined, then click **SSL inbound channel**. Set the SSL inbound channel properties to use the SSL configuration you defined (which requires client authentication). See Figure 20 on page 257.


Integrated Solutions Console Welcome w7admin

Help | Logout

View: All tasks

- Welcome
- Guided Activities
- Servers
  - Server Types
    - WebSphere application servers
    - WebSphere MQ servers
    - Web servers
- Applications
- Services
- Resources
- Security
- Environment
- System administration
- Users and Groups
- Monitoring and Tuning
- Troubleshooting
- Service integration
- UDDI

Cell=w7basea, Profile=default

Close page

Application servers

Application servers > w7sr00a > Transport Chain > SSL client authenticated > SSL inbound channel (SSL_4)

Use this page to view and configure a channel for handling the encryption and decryption of data over inbound connections.

Configuration

General Properties

* Transport channel name: SSL_4

Discrimination weight: 1

SSL Configuration

○ Centrally managed

● Specific to this endpoint

Select SSL Configuration: ClientAuthSSL

Apply OK Reset Cancel

Additional Properties

- Custom properties

Related Items

- SSL configuration - cell level
- View centrally managed SSL tree

Help

Field help

For field help information, select a field label or list marker when the help cursor is displayed.

Page help

More information about this page

Command Assistance

View administrative scripting command for last action

Figure 20. Setting the SSL inbound channel properties

9. Stop and restart the WebSphere server so that your changes take effect.

Results

When you are done, you set up a WebSphere SSL configuration that uses client authentication so that users can renew and revoke browser certificates.

Customizing the PKI Services web application

To customize the PKI Services web application, you need to perform the following tasks:

1. Update the template file.
2. (Optional) Modify the JSP files and update the EAR file with the modified JSP files.
3. Deploy the EAR file to a WebSphere Application Server

Updating the template file

You can customize the PKI Services web application by modifying the default copies of the template file, `pkitmpl.xml`. Any time that you update `pkitmpl.xml`, you need to create an equivalent copy of `pkixgen.tmpl`. You do this using the TemplateTool utility. For more information, see "Using the TemplateTool utility" on page 442.

(Optional) Modifying the JSP files and the EAR file

After you update the template file, you can optionally perform additional customization on the PKI Services web application by modifying the JSP files. You might want to modify the JSP files in the directory `mod_inc`. You can modify any line that is not marked with a comment saying that it cannot be modified. You also might want to modify JSP files in the directories `Customers` and `PKIServ`. You cannot modify JSP files in the directory `not_mod_inc`. For information about what files are contained in each directory, see "Locating JSP files for customizing web pages" on page 276.

Example: This example shows how you could customize a JSP file. `notbefore.jsp` is a JSP file in `mod_inc`. The portion of this file that displays the prompt and the options is shown in Figure 21 on page 258, without customization.

```
⋮
Number of days after today before the certificate becomes current<<%=optional_str %>

<SELECT NAME="notbefore">
<OPTION VALUE="0"
<% if (initvalue_str.equalsIgnoreCase("0")) out.print(" SELECTED "); %>
>0
<OPTION VALUE="30"
<% if (initvalue_str.equalsIgnoreCase("30")) out.print(" SELECTED "); %>
>30
</SELECT>
```

Figure 21. A portion of the JSP file `notbefore.jsp`, without customization

You could customize this code to reword the prompt and add an option of 7 days, as shown in Figure 22 on page 258.

```
⋮
Number of days after today before the certificate is valid<%=optional_str %>

<SELECT NAME="notbefore">
<OPTION VALUE="0"
<% if (initvalue_str.equalsIgnoreCase("0")) out.print(" SELECTED "); %>
>0

<OPTION VALUE="7"
<% if (initvalue_str.equalsIgnoreCase("7")) out.print(" SELECTED "); %>
>7

<OPTION VALUE="30"
<% if (initvalue_str.equalsIgnoreCase("30")) out.print(" SELECTED "); %>
>30
</SELECT>
```

Figure 22. A portion of the JSP file `notbefore.jsp`, with customization

An enterprise archive (EAR) file is a specialized Java archive (JAR) file, used to deploy Java EE applications to Java EE application servers. PKI Services ships a default EAR file in the directory `/usr/lpp/pkiserv/pkijsp`. If you make changes to the JSP files for the PKI Services web application, you need to update the EAR file to include your changes.

Steps for updating the EAR file using Traditional WebSphere Application Server

Perform the following steps to update the EAR file

Before you begin

You must have the `jar` command in your path. If you do not, define the JAVA_HOME variable (you can find its value on the WebSphere administration console) and add the `$JAVA_HOME/bin` directory to your path. For example:

```
export JAVA_HOME=/WebSphere/V6R1/AppServer/java
export PATH=$JAVA_HOME/bin:$PATH
```

Procedure

1. (Optional) Set up a directory to use just for updating the EAR file and make this your working directory. For example:

   ```
   cd $HOME
   mkdir pkiear
   cd pkiear
   ```

   ____

2. Copy the EAR file that you are using to the working directory. The default version that is shipped with PKI Services is in the file `PKIServ.ear` in the directory `/usr/lpp/pkiserv/pkijsp`. For example:

   ```
   cp /usr/lpp/pkiserv/pkijsp/PKIServ.ear .
   ```

   ____

3. Expand the EAR file by using the `jar` command. For example:

   ```
   jar -xvf PKIServ.ear
   ```

   ____

4. Expand the file `PKIServ_Web_war`:

   ```
   jar -xvf PKIServ_Web.war
   ```

   ____

5. Change to the modifiable include directory:

   ```
   cd mod_inc
   ```

   ____

6. The JSP files are in ASCII (ISO8859-1 code page). To edit them on z/OS, you must convert them to EBCDIC. Use `iconv` to convert a file to EBCDIC. For example, to convert the file `footer.jsp` to EBCDIC, enter:

   ```
   iconv -f iso8859-1 -t ibm-1047 footer.jsp > $HOME/footer.jsp.edit
   ```

   ____

7. Use `oedit` to edit the edit file that you created:

   ```
   oedit $HOME/footer.jsp.edit
   ```

   ____

8. Use `iconv` to convert the edited file back to ASCII. For example:

   ```
   iconv -t iso8859-1 -f ibm-1047 $HOME/footer.jsp.edit > footer.jsp
   ```

   ____

9. Go back to the directory containing the WAR and EAR files, and update the WAR file with the edited and reconverted JSP file:

   ```
   cd ..
   jar -uvf PKIServ_Web.war mod_inc/footer.jsp
   ```

   ____

10. Update the EAR file with the updated WAR file:

    ```
    jar -uvf PKIServ.ear PKIServ_Web.war
    ```

    ____

11. Make sure the `PKIServ.ear` file is publicly readable by issuing the `chmod` command.

    ```
    chmod 755 PKIServ.ear
    ```

    ____

Results

When you are done, you updated the JSP files, and updated the EAR file to include the updated JSP files. You can now deploy your updated EAR file to a WebSphere Application Server.

Steps for creating multiple applications
Perform the following steps to create multiple applications.

Before you begin

You need to have WebSphere administration skills. You might need to refer to information in the WebSphere Application Server Knowledge Center (www.ibm.com/support/knowledgecenter/SSEQTP).

Procedure

1. Copy over the WAR file to a new WAR file with a different name.

   ```
   cp PKIServ_Web.war PKIServ2_Web.war
   ```

   Then create a new EAR file with either a different name or in a new location.

   ```
   jar -cvf PKIServ_subca.ear PKIServ2_Web.war
   ```

2. Update `web.xml` to use the new domain.
3. Install the new EAR file through the console.

Steps for creating a new EAR alongside your existing EAR
Perform the following steps to create a new EAR file alongside your existing EAR file.

Before you begin

If you wish to keep your existing EAR file but also wish to add a second PKI application, you will need to transfer your customized files into a new sample EAR file.

Procedure

1. You will need a copy of the new sample EAR file. From the directory you wish to contain the modified EAR file:

   ```
   cp /usr/lpp/piserv/pkijsp/PKIServ.ear PKIServ.ear
   ```

   ____

2. Expand the new EAR and WAR files.

```
jar -xvf PKIServ.ear
jar -xvf PKIServ_Web.war
```

(Optional) If you wish to use a different *contextroot* name, rename your WAR file to that name at this point:

```
mv PKIServ_Web.war PKIServMod_Web.war
```

____

3. Replace the contents of the modifiable directory of the new EAR with the contents of your old EARS modifiable directory:

   ```
   rm <new directory>/mod_inc/*
   cp <old directory>/mod_inc/* <new directory>/mod_inc/*
   ```

4. Re-jar the WAR file. If you are using a new `contextroot`, **`jar`** to that WAR file name:

   ```
   jar -uvf PKIServMod_Web.war mod_inc/*
   ```

5. If you are not changing the `contextroot` you can use the original EAR file:

   ```
   jar -uvf PKIServ.ear PKIServ.Web.war
   ```

   Otherwise you must create a new one:

   ```
   jar -cvf PKIServMod.ear PKIServMod_Web.war
   ```

6. Deploy the new EAR file. Ensure that the isolated class loader box is checked to have multiple applications: **Environment > Shared Libraries > PKIServices > Class Loading**

Results

When you are done, you will have your original customized web pages with the ability to add multiple PKI Applications to your setup.

Deploying the EAR file to a WebSphere Application Server

To use the JSP files for the PKI Services web application, a WebSphere administrator must deploy them to a WebSphere Application Server.

Steps for deploying the EAR file to a Traditional WebSphere Application Server
Perform the following steps to deploy the EAR file to a WebSphere Application Server.

Before you begin

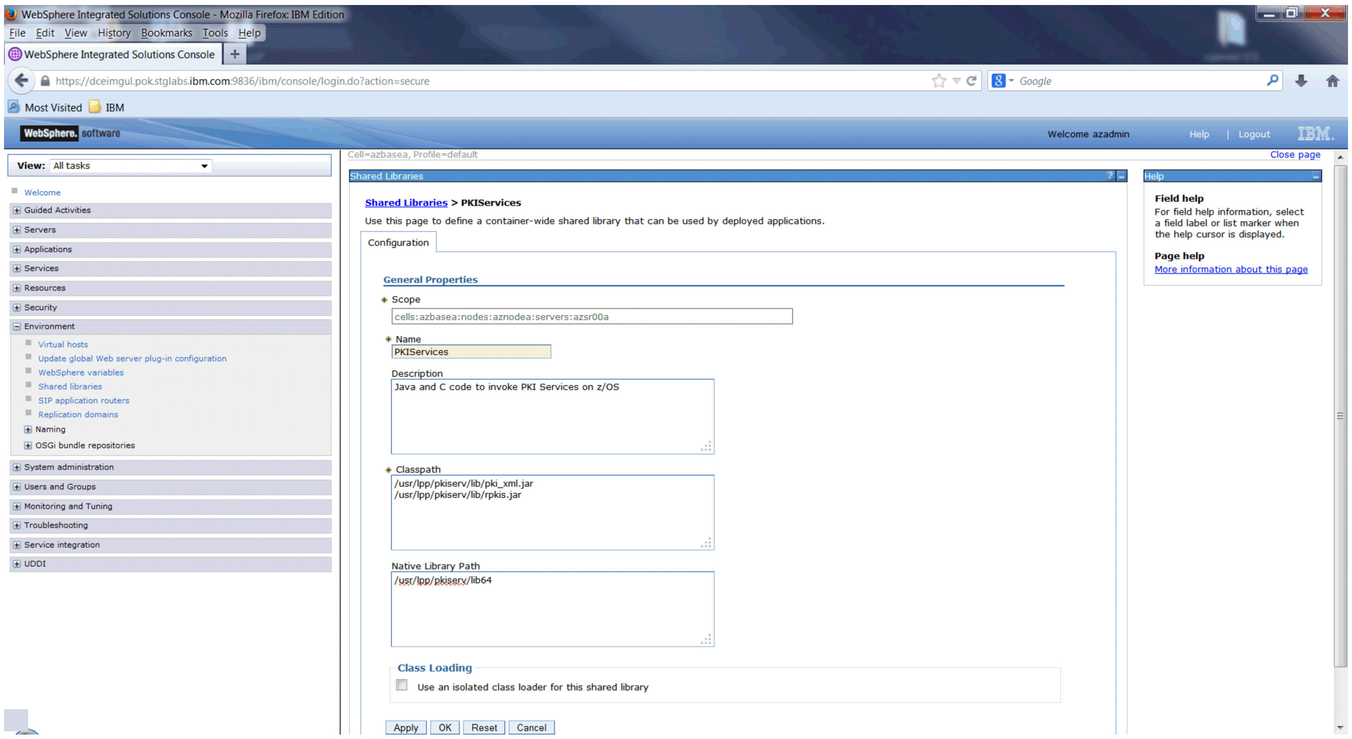
You need to have WebSphere administration skills. You might need to refer to information in the WebSphere Application Server Knowledge Center (www.ibm.com/support/knowledgecenter/SSEQTP).

Procedure

1. Log on to the WebSphere administrative console.

   ____

2. Create a new shared library.
   a) In the left pane, click **Environment > Shared Libraries**.
   b) Click **New** to add a shared library. The window shown in Figure 23 on page 262 opens.


WebSphere Integrated Solutions Console - Mozilla Firefox: IBM Edition

https://dceimgul.pokstglabs.ibm.com:9836/ibm/console/login.do?action=secure

Welcome azadmin | Help | Logout

View: All tasks

- Welcome
- Guided Activities
- Servers
- Applications
- Services
- Resources
- Security
- Environment
  - Virtual hosts
  - Update global Web server plug-in configuration
  - WebSphere variables
  - Shared libraries
  - SIP application routers
  - Replication domains
  - Naming
  - OSGi bundle repositories
- System administration
- Users and Groups
- Monitoring and Tuning
- Troubleshooting
- Service integration
- UDDI

Cell=azbasea, Profile=default

Shared Libraries

Shared Libraries > PKIServices

Use this page to define a container-wide shared library that can be used by deployed applications.

Configuration

General Properties

* Scope: cells:azbasea:nodes:aznodea:servers:azsr00a

* Name: PKIServices

Description: Java and C code to invoke PKI Services on z/OS

* Classpath:

```
/usr/lpp/pkiserv/lib/pki_xml.jar
/usr/lpp/pkiserv/lib/rpkis.jar
```

Native Library Path: /usr/lpp/pkiserv/lib64

Class Loading

☐ Use an isolated class loader for this shared library

Apply | OK | Reset | Cancel

Help

Field help
For field help information, select a field label or list marker when the help cursor is displayed.

Page help
More information about this page

Figure 23. The WebSphere Shared Libraries window

c) In the **Scope** field, select a scope that includes the server where you deploy the PKI Services JSPs (PKIServ_EAR).

d) In the **Name** field enter `PKIServices`.

e) Optionally enter a description in the **Description** field; for example, `Java and C code to invoke PKI Services on z/OS`.

f) In the **Classpath** field, enter the following two lines:

```
/usr/lpp/pkiserv/lib/pki_xml.jar
/usr/lpp/pkiserv/lib/rpkis.jar
```

g) In the **Native Library Path** field, enter:

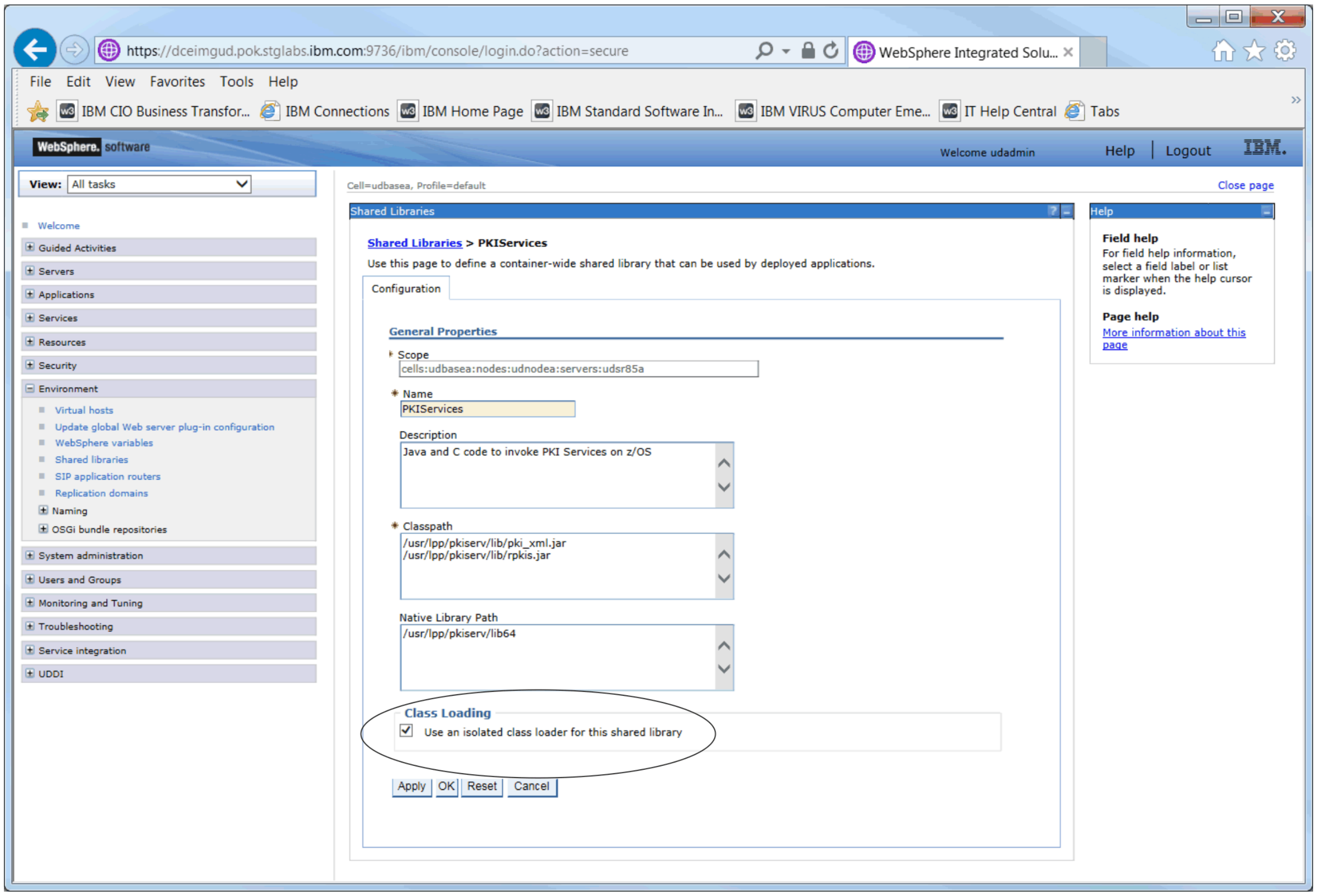
```
/usr/lpp/pkiserv/lib64
```

Note:

If running WebSphere Application Server in 31-bit mode, enter:

```
/usr/lpp/pkiserv/lib
```

h) Make sure to have the isolated class loader box checked to have multiple applications.


Shared Libraries > PKIServices

Use this page to define a container-wide shared library that can be used by deployed applications.

Configuration

General Properties

Scope: cells:udbasea:nodes:udnodea:servers:udsr85a

* Name: PKIServices

Description: Java and C code to invoke PKI Services on z/OS

* Classpath:
```
/usr/lpp/pkiserv/lib/pki_xml.jar
/usr/lpp/pkiserv/lib/rpkis.jar
```

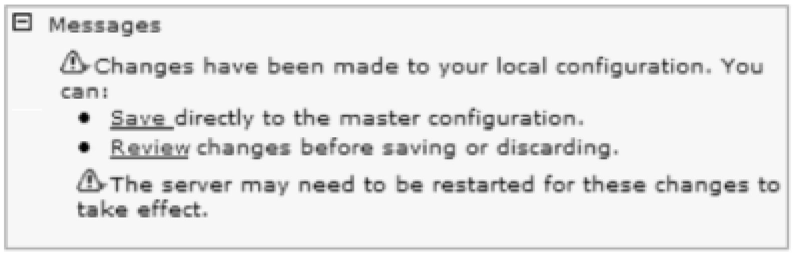
Native Library Path: /usr/lpp/pkiserv/lib64

Class Loading

☑ Use an isolated class loader for this shared library

Figure 24. The WebSphere Shared Libraries window with isolated class loader box checked.

i) Click **OK**. A message is displayed indicating that changes have been made to your local configuration. See Figure 25 on page 263.


Messages

Changes have been made to your local configuration. You can:

- Save directly to the master configuration.
- Review changes before saving or discarding.

The server may need to be restarted for these changes to take effect.

Figure 25. Message indicating that changes have been made to your local configuration.

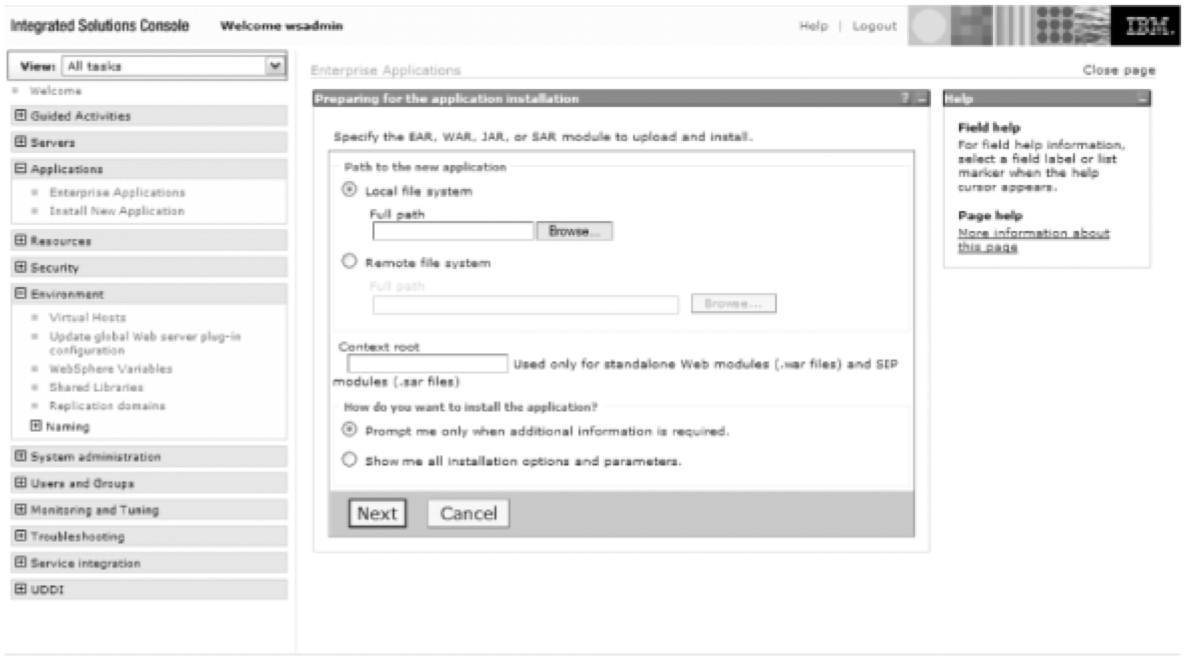
j) Click **Save** to save the changes to the master configuration.

____

3. Deploy the enterprise archive (EAR) file.

   a) In the left pane, click **Applications > Enterprise Applications**.

   b) Click **Install**. The window shown in Figure 26 on page 264 opens.



Specify the EAR, WAR, JAR, or SAR module to upload and install.

Path to the new application

(●) Local file system

Full path

( ) Remote file system

Full path

Context root

Used only for standalone Web modules (.war files) and SIP modules (.sar files)

How do you want to install the application?

(●) Prompt me only when additional information is required.

( ) Show me all installation options and parameters.

Field help

For field help information, select a field label or list marker when the help cursor appears.

Page help

More information about this page

Figure 26. Window for specifying the EAR file

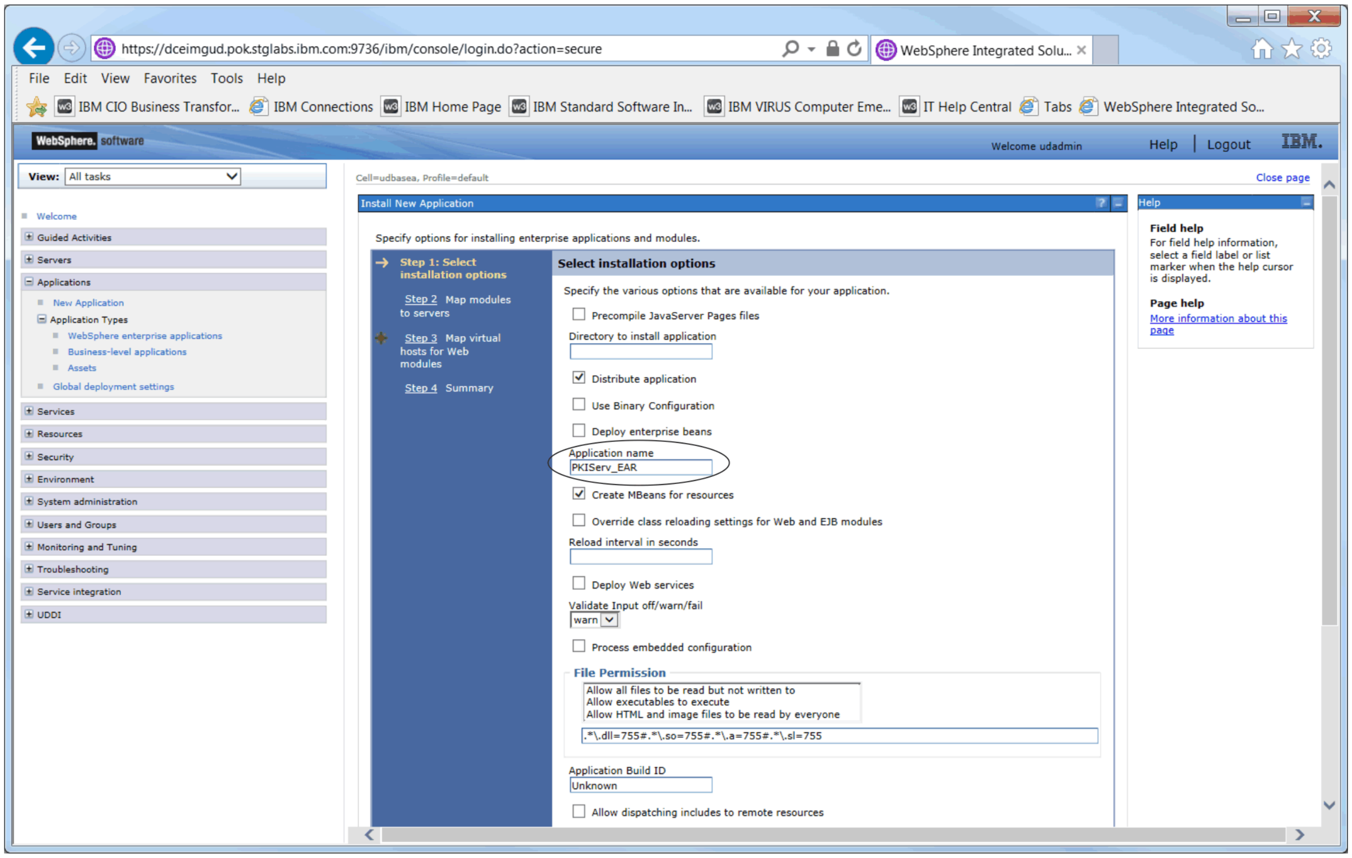
c) Click **Remote file system**, and click **Browse**.

d) Click the icon for the z/OS system. The root directory is displayed. Continue selecting until you reach:

```
$HOME/pkiear/PKIServ.ear
```

e) Click **OK** and **Next**.

f) If adding a second application, use a unique application name.


https://dceimgud.pokstglabs.ibm.com:9736/ibm/console/login.do?action=secure
WebSphere Integrated Solu...
File Edit View Favorites Tools Help
IBM CIO Business Transfor...
IBM Connections
IBM Home Page
IBM Standard Software In...
IBM VIRUS Computer Eme...
IT Help Central
Tabs
WebSphere Integrated So...
WebSphere. software
Welcome udadmin
Help
Logout
IBM.
View: All tasks
Welcome
Guided Activities
Servers
Applications
New Application
Application Types
WebSphere enterprise applications
Business-level applications
Assets
Global deployment settings
Services
Resources
Security
Environment
System administration
Users and Groups
Monitoring and Tuning
Troubleshooting
Service integration
UDDI
Close page
Help
Field help
For field help information, select a field label or list marker when the help cursor is displayed.
Page help
More information about this page

Cell=udbasea, Profile=default

Install New Application

Specify options for installing enterprise applications and modules.

Step 1: Select installation options
Step 2 Map modules to servers
Step 3 Map virtual hosts for Web modules
Step 4 Summary

Select installation options

Specify the various options that are available for your application.

- [ ] Precompile JavaServer Pages files

Directory to install application:

- [x] Distribute application
- [ ] Use Binary Configuration
- [ ] Deploy enterprise beans

Application name: PKIServ_EAR

- [x] Create MBeans for resources
- [ ] Override class reloading settings for Web and EJB modules

Reload interval in seconds:

- [ ] Deploy Web services

Validate Input off/warn/fail: warn

- [ ] Process embedded configuration

File Permission
Allow all files to be read but not written to
Allow executables to execute
Allow HTML and image files to be read by everyone
.*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755

Application Build ID: Unknown

- [ ] Allow dispatching includes to remote resources

Figure 27. The WebSphere Install New Application window.

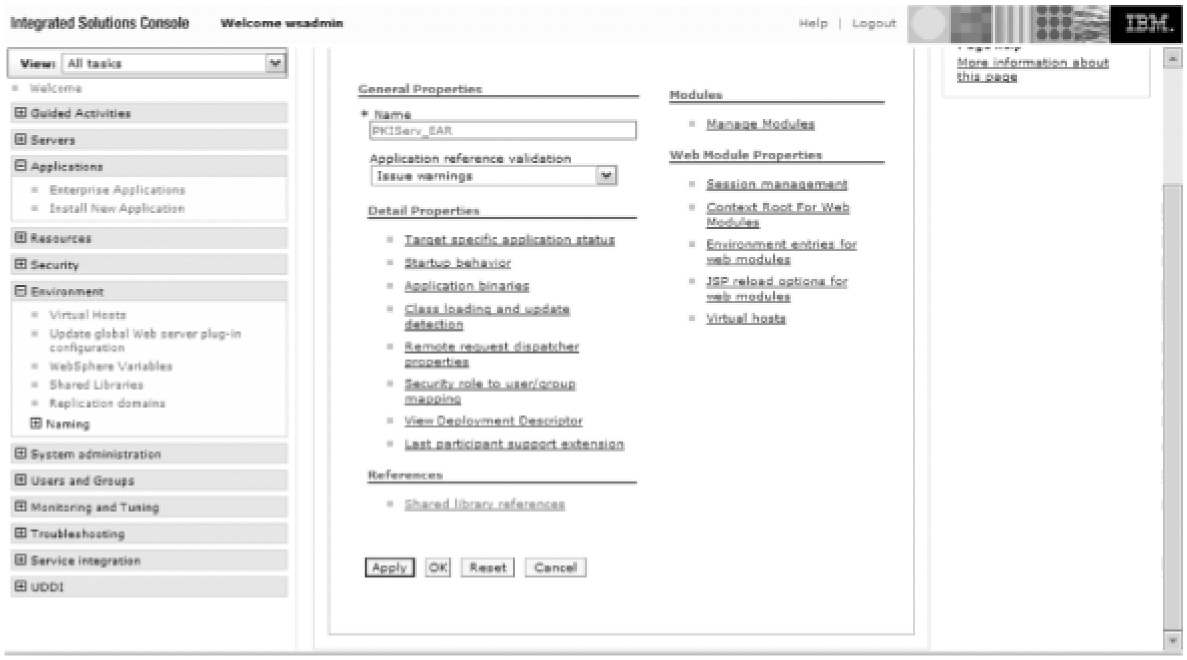
g) Continue clicking **OK** and **Next** or **Finish** until you see a message saying that the EAR was installed and the message shown in Figure 25 on page 263 indicating that changes have been made to your local configuration. (If you have more than one application installed into WebSphere Application Server, you might see a window where you must select a server to which you want to map modules before you click **Next**.)

h) Click **Save** to save the changes to the master configuration.

____

4. Associate the PKI Services shared library with the PKI Services application.

a) In the left pane, click **Applications > Enterprise Applications**. Click **PKIServ_EAR**. The window shown in Figure 28 on page 266 opens.


Integrated Solutions Console   Welcome wsadmin   Help | Logout

View: All tasks
- Welcome
- Guided Activities
- Servers
- Applications
  - Enterprise Applications
  - Install New Application
- Resources
- Security
- Environment
  - Virtual Hosts
  - Update global Web server plug-in configuration
  - WebSphere Variables
  - Shared Libraries
  - Replication domains
  - Naming
- System administration
- Users and Groups
- Monitoring and Tuning
- Troubleshooting
- Service integration
- UDDI

General Properties

* Name: PKIServ_EAR

Application reference validation: Issue warnings

Detail Properties
- Target specific application status
- Startup behavior
- Application binaries
- Class loading and update detection
- Remote request dispatcher properties
- Security role to user/group mapping
- View Deployment Descriptor
- Last participant support extension

References
- Shared library references

Apply   OK   Reset   Cancel

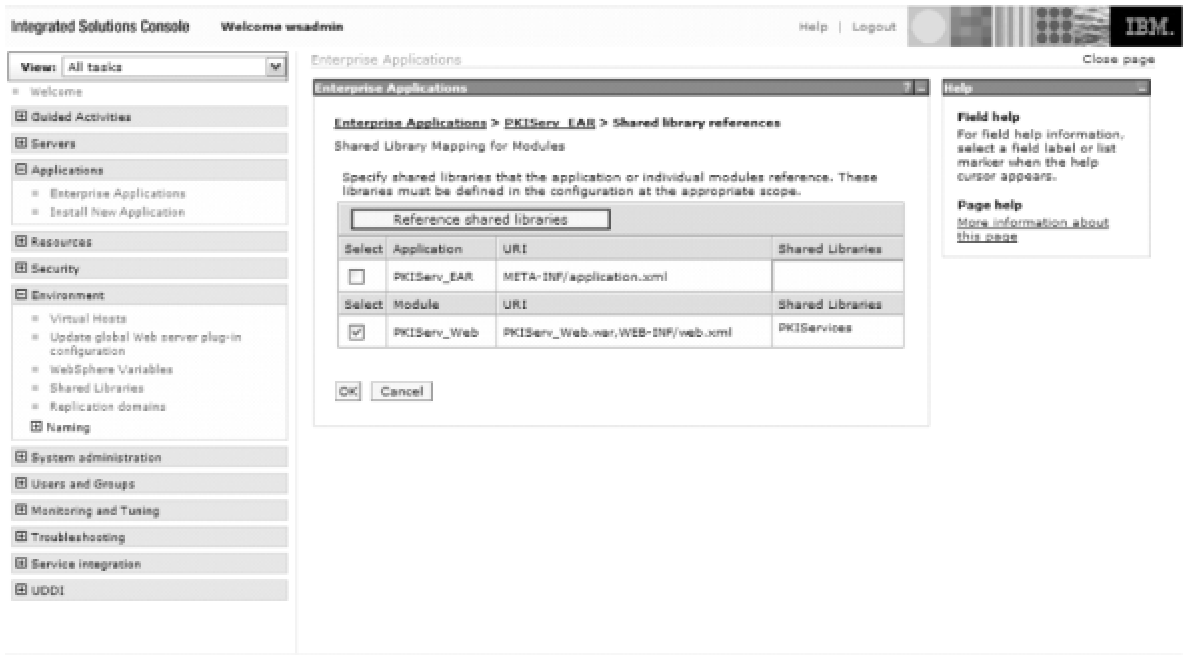
Modules
- Manage Modules

Web Module Properties
- Session management
- Context Root For Web Modules
- Environment entries for web modules
- JSP reload options for web modules
- Virtual hosts

More information about this page

Figure 28. Application properties page

b) Click **Shared Library References**. The window shown in Figure 29 on page 266 opens.


Integrated Solutions Console   Welcome wsadmin   Help | Logout

View: All tasks
- Welcome
- Guided Activities
- Servers
- Applications
  - Enterprise Applications
  - Install New Application
- Resources
- Security
- Environment
  - Virtual Hosts
  - Update global Web server plug-in configuration
  - WebSphere Variables
  - Shared Libraries
  - Replication domains
  - Naming
- System administration
- Users and Groups
- Monitoring and Tuning
- Troubleshooting
- Service integration
- UDDI

Enterprise Applications   Close page

Enterprise Applications

Enterprise Applications > PKIServ_EAR > Shared library references

Shared Library Mapping for Modules

Specify shared libraries that the application or individual modules reference. These libraries must be defined in the configuration at the appropriate scope.

Reference shared libraries

Select	Application	URI	Shared Libraries
☐	PKIServ_EAR	META-INF/application.xml	
Select	**Module**	**URI**	**Shared Libraries**
☑	PKIServ_Web	PKIServ_Web.war,WEB-INF/web.xml	PKIServices

OK   Cancel

Help

Field help
For field help information, select a field label or list marker when the help cursor appears.

Page help
More information about this page

Figure 29. Shared library mapping for modules page

c) Select the **PKIServ_Web** check box and click **Reference Shared Library**.

d) Add the PKI Services shared library to the Selected list and click OK.

e) Select the **PKIServ_EAR** check box and click **Reference Shared Library**.

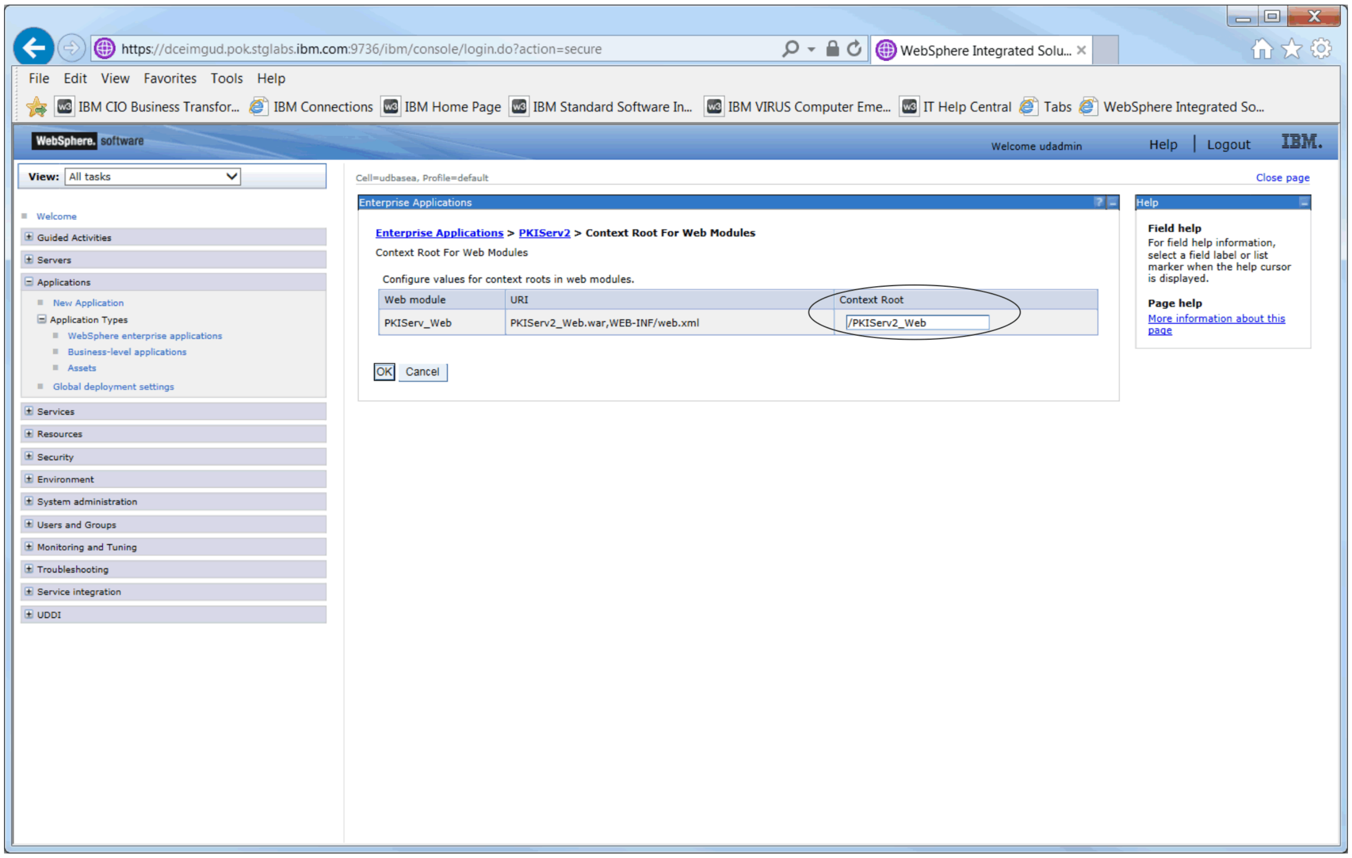
f) Add the PKI Services shared library to the Selected list and click OK.

g) Again you see the message shown in Figure 25 on page 263 indicating that changes have been made to your local configuration. Click **Save** to save the changes to the master configuration.

____

5. If you are using multiple applications, check that the context root matches the unique WAR file name:

In the left panel, click **Applications > Enterprise Applications**. Click the name of your application. Click **Context Root for Web Modules**.



Cell=udbasea, Profile=default

Enterprise Applications > PKIServ2 > **Context Root For Web Modules**

Context Root For Web Modules

Configure values for context roots in web modules.

Web module	URI	Context Root
PKIServ_Web	PKIServ2_Web.war,WEB-INF/web.xml	/PKIServ2_Web

Field help
For field help information, select a field label or list marker when the help cursor is displayed.

Page help
More information about this page

https://dceimgud.pok.stglabs.ibm.com:9736/ibm/console/login.do?action=secure
WebSphere Integrated Solu...
File Edit View Favorites Tools Help
IBM CIO Business Transfor... IBM Connections IBM Home Page IBM Standard Software In... IBM VIRUS Computer Eme... IT Help Central Tabs WebSphere Integrated So...
WebSphere. software
Welcome udadmin
Help | Logout
IBM.
View: All tasks
Welcome
Guided Activities
Servers
Applications
New Application
Application Types
WebSphere enterprise applications
Business-level applications
Assets
Global deployment settings
Services
Resources
Security
Environment
System administration
Users and Groups
Monitoring and Tuning
Troubleshooting
Service integration
UDDI
Close page
Enterprise Applications
Help
OK
Cancel

Figure 30. The WebSphere Enterprise Applications Context Root for Web Modules.

The context root can be updated at this time if necessary.

Results

When you are done, you have deployed the PKI Services JSP files to a WebSphere Application Server. You can now begin using the customized PKI Services web application.

Roadmap for implementing the PKI Services web application using JSPs in Liberty

About this task

To implement the PKI Services web application using JSPs in Liberty, perform the tasks in Table 45 on page 267.

Table 45. Task roadmap for implementing the PKI Services web application using the JSPs in Liberty

Subtask	Associated instructions (see . . .)
Preparation	"Steps for preparing to implement the PKI Services web application using JSPs in Liberty" on page 268
Give Liberty users authorization to use PKI Services functions	"Giving Liberty users authorization to use PKI Services functions" on page 268

Table 45. Task roadmap for implementing the PKI Services web application using the JSPs in Liberty (continued)

Subtask	Associated instructions (see . . .)
Customize the web application	"Customizing the PKI Services web application" on page 273
• Update the template file.	• "Updating the template file" on page 258
• (Optional) Modify the JSP files and update the EAR file with the modified JSP files.	• "(Optional) Modifying the JSP files and the EAR file" on page 258

Steps for preparing to implement the PKI Services web application using JSPs in Liberty

About this task

While Traditional WebSphere Application Server uses administrative console to configure authorization, configuration in Liberty is mainly done through the `server.xml` file. Perform the following steps to prepare to implement the PKI Services web application using JSPs.

Procedure

1. Set the _PKISERV_ENABLE_JSP environment variable to indicate that you are using the JSPs instead of the REXX CGIs. To do this, uncomment the following line in the environment variables file `pkiserv.envars`:

   ```
   _PKISERV_ENABLE_JSP=TRUE
   ```

   (If the line is not in your copy of the environment variables file, add it.)
2. If you have not previously done so, copy the XML template file and the XML schema file from the directory in which the MVS programmer installed PKI Services to the runtime directory by entering the following commands from the UNIX command line. The default directories are `/usr/lpp/pkiserv/` and `/etc/pkiserv` respectively. The user ID you use for copying files must have superuser authority.

   ```
   cp -p /install-dir/samples/pkitmpl.xml runtime-dir
   cp -p /install-dir/samples/PKIServ.xsd runtime-dir
   ```

Results

Note: You do not copy the file `pkixgen.tmpl`. PKI Services does not ship a copy of this file in the `samples` directory. You generate `pkixgen.tmpl` from `pkitmpl.xml` after you customize `pkitmpl.xml`.

When you are done, you are ready to customize the PKI Services web application.

Giving Liberty users authorization to use PKI Services functions

You need to give each Liberty user that uses the PKI Services web application authorization to use PKI Services functions.

Steps for giving Liberty users authorization to use PKI Services functions

About this task

Perform the following steps to give Liberty users authorization to use PKI Services functions.

Procedure

1. You may either work with the generated `server.xml` file when you created your liberty server, or use the shipped sample file.

If you are using the shipped sample, copy over `server.xml` from the samples directory. This file will initially be in EBCDIC format:

```
cp /usr/lpp/pkiserv/samples/server.xml <libertyServerLocation>/server.xml.ebcdic
```

The file needs to be converted into ASCII format before running it. Convert the file into ASCII format under the name `server.xml`:

```
iconv -t iso8859-1 -f ibm-1047 server.xml.ebcdic > server.xml
```

You must then tag the file so that it is in readable format:

```
chtag -tc ISO8859-1 server.xml
```

Note: In order for the proper automatic conversion of the tagged file to occur, your _BPXK_AUTOCVT UNIX System Services environmental value must be set to ON.

If you are not using the sample file that is provided, include the following lines in your `server.xml`:

```
<featureManager>
 <feature>jsp-2.2</feature>
 <feature>localConnector-1.0</feature>
 <feature>ssl-1.0</feature>
 <feature>appSecurity-2.0</feature>
 <feature>servlet-3.0</feature>
 <feature>ejbLite-3.1</feature>
 <feature>zosSecurity-1.0</feature>
</featureManager>

<applicationMonitor updateTrigger="mbean"/>

<safRegistry id="saf"/>

<syncToOSThread appEnabled="true"/>

<safCredentials profilePrefix="BBGZDFLT"/>


<safAuthorization enableDelegation="true"/>

<webAppSecurity ssoRequireSSL='true'/>

<httpSession cookieSecure='true'/>

<library id="global">
   <fileset dir="/usr/lpp/pkiserv/lib" includes="*.jar"
         scanInterval="5s"/>
   <fileset dir="/usr/lpp/pkiserv/lib64" includes="*.so"
         scanInterval="5s"/>
</library>
```

2. The `web.xml` file for PKIServ_Web defines three roles: SAFuser, PKIAdmin and PKISurrogate. You need the SAFuser role only if you want to specify z/OS authentication (either zAuthRunAsSurrogate or zAuthRunAsClient) for certificate request or retrieval authentication. In the sample `pkitmpl.xml` file, z/OS authentication is used for these certificate templates: 1-Year SAF Browser Certificate, 1-Year SAF Server Certificate, 2-Year PKI Browser Certificate For Authenticating To z/OS, 5-Year SCEP Certificate - Preregistration, 2-Year EST Certificate Preregistration, and 5-Year PKI Intermediate CA Certificate. The PKIAdmin role allows users assigned it to use the PKI Services administration web pages.

   If you are using SAF authorization, roles are mapped to EJBROLE resource profiles. In order to be considered as eligible for a role, a user must have read access to the EJBROLE profile or must be connected to a SAF group that has read access. The corresponding SAF resource profile for a given application and role is named `{`*`profilePrefix`*`}.{`*`appName`*`}.{`*`roleName`*`}` in the EJBROLE SAF class. `{`*`profilePrefix`*`}` is specified by the `profilePrefix` attribute in the `<safCredentials>` configuration element. If you do not specify this element, then the default `profilePrefix` of BBGZDFLT is used. `{`*`appName`*`}` is specified by the name attribute in the `<enterpriseApplication>` configuration element. A unique name attribute value should be specified for each `<enterpriseApplication>` configuration element. If you do not specify the name attribute, the name will be generated from the base name of the ear file specified in the location attribute. PKIServ_root.*{roleName}* is one of either PKIAdmin, SAFuser, or PKISurrogate as appropriate.

To create the SAFuser and PKIAdmin roles using SAF EJBROLE class profiles, enter the following TSO commands. The first RDEFINE command creates the SAFuser role and gives it to all users who are authenticated. The second RDEFINE command creates the PKIAdmin role.

```
SETROPTS CLASSACT(EJBROLE)
RDEFINE EJBROLE BBGZDFLT.PKI.SAFuser UACC(READ)
RDEFINE EJBROLE BBGZDFLT.PKI.PKIAdmin UACC(NONE)
```

Then enter *one* of the following commands. The first PERMIT command must be entered for *each* user who should have authorization to use the PKI Services administration web pages.

```
PERMIT BBGZDFLT.PKI.PKIAdmin CLASS(EJBROLE) ID(userid) ACCESS(READ)
```

or

```
PERMIT BBGZDFLT.PKI.PKIAdmin CLASS(EJBROLE) ID(pkigrp) ACCESS(READ)
```

where *pkigrp* is the value of the variable pkigroup in the IKYSETUP exec (default value PKIGRP). For a description of the pkigroup variable, see Table 19 on page 55.

For SAF delegation for the PKISurrogate role, create a SAF EJBROLE class profile with the surrogate ID specified in the APPLDATA. Change the value PKISERV to the user ID you specified for the surrogate user in the IKYSETUP exec, if you specified a different user id. (See see Table 19 on page 55).

```
RDEFINE EJBROLE BBGZDFLT.PKI.PKISurrogate UACC(READ) APPLDATA('PKISERV')
```

Refresh the EJBROLE class after all the profile updates

```
SETROPTS RACLIST(EJBROLE) REFRESH
```

Grant the user ID of the Liberty Server CONTROL access to the BBG.SYNC.BBGZDFLT profile in the FACILITY class. This access allows the server to sync any RunAs identity with the OS identity:

```
RDEFINE FACILITY BBG.SYNC.BBGZDFLT UACC(NONE)
PERMIT BBG.SYNC.BBGZDFLT ID(LIBSVR) ACCESS(CONTROL) CLASS(FACILITY)
SETROPTS RACLIST(FACILITY) REFRESH
```

The preceding class profiles take effect if you have SAF authorization with delegation in server.xml not commented.

```
<safAuthorization enableDelegation="true" />
```

3. Issue TSO commands to give Liberty users authorization to use PKI Services.

   a) Give the userID for the Liberty server READ access to the FACILITY class profile IRR.RPKISERV.<function>.<ca-domain>. For unnamed domain issue the command:

   ```
   PERMIT IRR.RPKISERV.* CLASS(FACILITY) ID(LIBSVR) ACCESS(READ)
   ```

   For named domain issue the command:

   ```
   PERMIT IRR.RPKISERV.*.<DomainName> CLASS(FACILITY) ID(LIBSVR) ACCESS(READ)
   ```

   b) Authorize the PKI Services surrogate user to use the R_PKIServ functions REQCERT, GENCERT, ADD, VERIFY, REVOKE, RESPOND, SCEPREQ, REQRENEW, GENRENEW, EXPORT, and QRECOVER. Enter the following TSO commands. Change the value PKISERV to the user ID you specified for the surrogate user in the IKYSETUP exec, if you specified a different user ID. (See Table 19 on page 55.)

   ```
   RDEFINE FACILITY IRR.DIGTCERT.REQCERT UACC(NONE)
   PERMIT IRR.DIGTCERT.REQCERT CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

   RDEFINE FACILITY IRR.DIGTCERT.GENCERT UACC(NONE)
   ```

```
PERMIT IRR.DIGTCERT.GENCERT CLASS(FACILITY) ID(PKISERV) ACCESS(CONTROL)

RDEFINE FACILITY IRR.DIGTCERT.ADD UACC(NONE)
PERMIT IRR.DIGTCERT.ADD CLASS(FACILITY) ID(PKISERV) ACCESS(UPDATE)

RDEFINE FACILITY IRR.DIGTCERT.VERIFY UACC(NONE)
PERMIT IRR.DIGTCERT.VERIFY CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

RDEFINE FACILITY IRR.DIGTCERT.REVOKE UACC(NONE)
PERMIT IRR.DIGTCERT.REVOKE CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

RDEFINE FACILITY IRR.DIGTCERT.RESPOND UACC(NONE)
PERMIT IRR.DIGTCERT.RESPOND CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

RDEFINE FACILITY IRR.DIGTCERT.SCEPREQ UACC(NONE)
PERMIT IRR.DIGTCERT.SCEPREQ CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

RDEFINE FACILITY IRR.DIGTCERT.REQRENEW UACC(NONE)
PERMIT IRR.DIGTCERT.REQRENEW CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

RDEFINE FACILITY IRR.DIGTCERT.GENRENEW UACC(NONE)
PERMIT IRR.DIGTCERT.GENRENEW CLASS(FACILITY) ID(PKISERV) ACCESS(READ)

RDEFINE FACILITY IRR.DIGTCERT.EXPORT UACC(NONE)
PERMIT IRR.DIGTCERT.EXPORT CLASS(FACILITY) ID(PKISERV) ACCESS(CONTROL)

RDEFINE FACILITY IRR.DIGTCERT.QRECOVER UACC(NONE)
PERMIT IRR.DIGTCERT.QRECOVER CLASS(FACILITY) ID(PKISERV) ACCESS(READ)
```

c) If you modified `pkitmpl.xml` and use the zAuthRunAsClient retrieve authentication type:

```
<tns:retrieve_authtype>zAuthRunAsClient</tns:retrieve_authtype>
```

you must give any user ID that requests this kind of certificate authorization to the R_PKIServ EXPORT function:

```
PERMIT IRR.DIGTCERT.EXPORT CLASS(FACILITY) ID(userid) ACCESS(CONTROL)
```

d) Refresh the in-storage profiles so that the changes you made to the FACILITY class take effect:

```
SETR RACLIST(FACILITY) REFRESH
```

Results
When you are done, you authorized Liberty users to use PKI Services functions.

Setting up the key ring for Liberty users to access the PKI Services web pages

Steps for setting up the key ring
If you want to use an existing key ring and server certificate for the Liberty server, you may skip the steps within section **A**.

Before you begin

Procedure

A) Set up key ring and server certificate

1. Create a RACF key ring for the Liberty server ID (LIBSVR in this example):

```
RACDCERT ADDRING(LibertyKeyring) ID(LIBSVR)
```

2. Connect the PKI CA certificate to this key ring. The CA certificate label ("Local PKI CA") is defined in the IKYSETUP exec.

```
RACDCERT ID(LIBSVR) CONNECT(CERTAUTH LABEL('Local PKI CA')
RING(LibertyKeyring))
```

3. Create a server certificate that is signed by the PKI CA certificate with the SUBJECTSDN matching the httpEndPoint host specified in server.xml, and connect this server certificate to the key ring created in Step 1.

```
RACDCERT GENCERT
   ID(LIBSVR) SIGNWITH(CERTAUTH LABEL('Local PKI CA'))
   WITHLABEL('SSL Cert') SUBJECTSDN(CN(<www.YourCompany.com>)
   O(<Your Company>) L(<Your City>)
   SP(<Your Full State or Province Name>)
   C(<Your Country 2 Letter Abbreviation>))

RACDCERT ID(LIBSVR)
   CONNECT(ID(LIBSVR) LABEL('SSL Cert') RING(LibertyKeyring)
   USAGE(PERSONAL) DEFAULT)
```

B) Configure server.xml

4. Specify the location of key ring.

   If you create the new key ring:

```
location="safkeyring://libsvr/LibertyKeyring"
```

   If you use the existing server certificate and the key ring, for example SSLring owned by WEBSRV:

```
location="safkeyring://websrv/SSLring"
```

   Note: Specify your keyring location and a password value of password.

```
<keyStore id="defaultKeyStore" location="safkeyring://libsvr/LibertyKeyring"
type="JCERACFKS" password="password" fileBased="false" readOnly="true"/>
```

5. In the trustStoreRef and keyStoreRef fields, enter the name of the keystore that you just defined:

```


<ssl id="ServerAuthSSLConfig"
     trustStoreRef="defaultKeyStore"
     keyStoreRef="defaultKeyStore"
     clientAuthentication="false"/>

<ssl id="ClientAuthConfig"
     trustStoreRef="defaultKeyStore"
     keyStoreRef="defaultKeyStore"
     clientAuthentication="true"
     sslProtocol="TLS"/>
```

6. Specify your host name and ports for server authentication and client authentication.

   Note: The httpPort values should match those in your web.xml

```
<httpEndpoint host="pkiserver.mycompany.com"
 httpPort="9080" httpsPort="9443" id="defaultHttpEndpoint">
 <sslOptions sslRef="ServerAuthSSLConfig"/>
</httpEndpoint>

<httpEndpoint host="pkiserver.mycompany.com"
 httpsPort="9444" id="secondHttpEndpoint">
 <sslOptions sslRef="ClientAuthConfig"/>
</httpEndpoint>
```

C) Authorize the Liberty server to access the server certificate and its private key in the key ring

7. If you create a new server certificate and a new key ring:

```
SETROPTS CLASSACT(RDATALIB)
RDEFINE RDATALIB LIBSVR.LibertyKeyring.LST UACC(NONE)
PERMIT LIBSVR.LibertyKeyring.LST CLASS(RDATALIB) ID(LIBSVR) ACCESS(READ)
SETROPTS RACLIST(RDATALIB) REFRESH
```

If you use the existing server certificate and the key ring, for example `SSLring` owned by `WEBSVR`:

```
SETROPTS CLASSACT(RDATALIB)
RDEFINE RDATALIB WEBSVR.SSLring.LST UACC(NONE)
PERMIT WEBSRV.SSLring.LST ID(LIBSVR) ACCESS(UPDATE)
SETROPTS RACLIST(RDATALIB) REFRESH
```

D) The CA root certificate of the Liberty server must be distributed to the users who require access to the PKI web page interfaces. A possible method is to distribute the CA root certificate by using the same communication channel that you use to provide the users with the URI of the PKI web page. For example, if you send email, you can:

8. Add the CA root certificate as an attachment or include it as Base64 encoded text.
9. Send a separate communication with the root certificate fingerprint to help users ensure that the correct CA certificate was received in the previous email.

Results

When you are done, you set up a Liberty SSL configuration for both server and client authentication. Client authentication is required for users to renew and revoke browser certificate.

Customizing the PKI Services web application

To customize the PKI Services web application, you need to perform the following tasks:

1. Update the template file.
2. (Optional) Modify the JSP files and update the EAR file with the modified JSP files.
3. Make sure the name of the EAR file is specified with the full path in the location field in the `enterpriseApplication` section in `server.xml`.

Updating the template file

You can customize the PKI Services web application by modifying the default copies of the template file, `pkitmpl.xml`. Any time that you update `pkitmpl.xml`, you need to create an equivalent copy of `pkixgen.tmpl`. You do this using the TemplateTool utility. For more information, see "Using the TemplateTool utility" on page 442.

(Optional) Modifying the JSP files and the EAR file

After you update the template file, you can optionally perform additional customization on the PKI Services web application by modifying the JSP files. You might want to modify the JSP files in the directory `mod_inc`. You can modify any line that is not marked with a comment saying that it cannot be modified. You also might want to modify JSP files in the directories `Customers` and `PKIServ`. You cannot modify JSP files in the directory `not_mod_inc`. For information about what files are contained in each directory, see "Locating JSP files for customizing web pages" on page 276.

Example: This example shows how you could customize a JSP file. `notbefore.jsp` is a JSP file in `mod_inc`. The portion of this file that displays the prompt and the options is shown in Figure 31 on page 273, without customization.

```
⋮
Number of days after today before the certificate becomes current<<%=optional_str %>

<SELECT NAME="notbefore">
<OPTION VALUE="0"
<% if (initvalue_str.equalsIgnoreCase("0")) out.print(" SELECTED "); %>
>0
<OPTION VALUE="30"
<% if (initvalue_str.equalsIgnoreCase("30")) out.print(" SELECTED "); %>
>30
</SELECT>
```

Figure 31. A portion of the JSP file `notbefore.jsp`, without customization

You could customize this code to reword the prompt and add an option of 7 days, as shown in Figure 32 on page 274.

```
⋮
Number of days after today before the certificate is valid<%=optional_str %>

<SELECT NAME="notbefore">
<OPTION VALUE="0"
<% if (initvalue_str.equalsIgnoreCase("0")) out.print(" SELECTED "); %>
>0

<OPTION VALUE="7"
<% if (initvalue_str.equalsIgnoreCase("7")) out.print(" SELECTED "); %>
>7

<OPTION VALUE="30"
<% if (initvalue_str.equalsIgnoreCase("30")) out.print(" SELECTED "); %>
>30
</SELECT>
```

Figure 32. A portion of the JSP file `notbefore.jsp`, with customization

An enterprise archive (EAR) file is a specialized Java archive (JAR) file, used to deploy Java EE applications to Java EE application servers. PKI Services ships a default EAR file in the directory `/usr/lpp/pkiserv/pkijsp`. If you make changes to the JSP files for the PKI Services web application, you need to update the EAR file to include your changes.

Steps for updating the EAR file

Perform the following steps to update the EAR file.

Before you begin

Note: Steps 1 - 11 are the same as for updating the EAR file for Traditional WebSphere Application Server.

You must have the `jar` command in your path. If you do not, define the JAVA_HOME variable and add the `$JAVA_HOME/bin` directory to your path. (Installation specific values such as JAVA_HOME are defined as Environment Variables available through the WebSphere administration console.) For example:

```
export JAVA_HOME=/WebSphere/AppServer/java
export PATH=$JAVA_HOME/bin:$PATH
```

Procedure

1. (Optional) Set up a directory to use just for updating the EAR file and make this your working directory. For example:

   ```
   cd $HOME
   mkdir pkiear
   cd pkiear
   ```

   ____

2. Copy the EAR file that you are using to the working directory. The default version that is shipped with PKI Services is in the file `PKIServ.ear` in the directory `/usr/lpp/pkiserv/pkijsp`. For example:

   ```
   cp /usr/lpp/pkiserv/pkijsp/PKIServ.ear .
   ```

   ____

3. Expand the EAR file by using the `jar` command. For example:

   ```
   jar -xvf PKIServ.ear
   ```

   ____

4. Expand the file PKIServ_Web_war:

```
jar -xvf PKIServ_Web.war
```

5. Change to the modifiable include directory:

```
cd mod_inc
```

6. The JSP files are in ASCII (ISO8859-1 code page). To edit them on z/OS, you must convert them to EBCDIC. Use iconv to convert a file to EBCDIC. For example, to convert the file footer.jsp to EBCDIC, enter:

```
iconv -f iso8859-1 -t ibm-1047 footer.jsp > $HOME/footer.jsp.edit
```

7. Use oedit to edit the edit file that you created:

```
oedit $HOME/footer.jsp.edit
```

8. Use iconv to convert the edited file back to ASCII. For example:

```
iconv -t iso8859-1 -f ibm-1047 $HOME/footer.jsp.edit > footer.jsp
```

9. Go back to the directory containing the WAR and EAR files, and update the WAR file with the edited and reconverted JSP file:

```
cd ..
jar -uvf PKIServ_Web.war mod_inc/footer.jsp
```

10. Update the EAR file with the updated WAR file:

```
jar -uvf PKIServ.ear PKIServ_Web.war
```

11. Make sure the PKIServ.ear file is publicly readable by issuing the chmod command.

```
chmod 755 PKIServ.ear
```

12. Update your server.xml file to point to the location of your modified EAR file.

```
<enterpriseApplication id="PKIServ_Web"
location="$HOME/pkiear/PKIServ.ear" name="PKIServ_Web">
```

Results

When you are done, you updated the JSP files, and updated the EAR file to include the updated JSP files.

Steps for retrofitting release changes into the PKI Services certificate templates and JSPs

If you used an earlier release of PKI Services, you might need to retrofit changes in the `pkitmpl.xml` certificate templates file. (You would not want to replace the file if you customized it in the previous release.).

For retrofitting release changes into the templates file, review the following information.

You can use a file comparison tool to compare the new PKI Services certificates template file (`/usr/lpp/pkiserv/samples/pkitmpl.xml`) and your existing PKI Services certificates template file (`/etc/pkiserv/pkitmpl.xml`).

Perform the following steps to retrofit changes into the `pkitmpl.xml` certificate templates file so you do not lose any customization that you made in a previous release.

1. Make a backup copy of your current certificate templates file. For example, enter from the UNIX command line:

   ```
   cp /etc/pkiserv/pkitmpl.xml /etc/pkiserv/pkitmpl.backup
   ```

   ____
2. Copy the new sample templates file to the runtime location. (This is the copy that you edit.)

   ```
   cp /usr/lpp/pkiserv/samples/pkitmpl.xml /etc/pkiserv/pkitmpl.xml
   ```

   ____
3. Using a compare program of your choice, compare the two template files:
   - `/etc/pkiserv/pkitmpl.xml`
   - `/etc/pkiserv/pkitmpl.backup`

   ____
4. Edit the runtime copy of the templates file (`/etc/pkiserv/pkitmpl.xml`). Using the compare output that is generated in Step 3, merge the changes that you made to the original template file into the runtime copy of the templates file.

   ____
5. Exit the file to save your changes.

   ____

For retrofitting release changes into JSP file, review the following information.

You can use a file comparison tool to compare the new JSP files and your existing PKI Services JSP files. For more information, see "Locating JSP files for customizing web pages" on page 276.

For each JSP file, follow the preceding steps that are indicated for the pkitmpl.xml file.

Locating JSP files for customizing web pages

The JSP files are shipped in `PKIServ_Web.war` in `PKIServ.ear` and are in the following directories:

- `Customers`

  JSPs used by end users. Each JSP file in this directory defines all HTML and script for an entire end-user web page. Each file starts with an <HTML> tag and ends with a </HTML> tag. JSP files in this directory can be modified, but contain lines of Java processing that cannot be modified. Lines that cannot be modified are identified with comments. Table 46 on page 277 lists the JSP files in this directory.
- `PKIServ`

JSPs used by administrators. Each JSP file in this directory defines all HTML and script for an entire administrator web page. Each file starts with an <HTML> tag and ends with a </HTML> tag. JSP files in this directory can be modified, but contain lines of Java processing that cannot be modified. Lines that cannot be modified are identified with comments. Table 47 on page 278 lists the JSP files in this directory.

- `mod_inc`

  Modifiable include files. Each JSP file in this directory defines HTML and script (JavaScript) that might be used on multiple web pages, such as running headers or footers for a page, or a form field element (including JavaScript to verify the input). (If you are familiar with the text template file, `pkiserv.tmpl`, used with the REXX CGIs, these modifiable include files generally correspond to an INSERT section.) Installations are likely to want to customize the code in these files. Some parts of these files cannot be modified and these parts are identified by comments in the file. Table 48 on page 279 lists the JSP files in this directory. "Customizing the PKI Services web application" on page 257 shows an example of customizing a JSP file in this directory.

- `not_mod_inc`

  Non-modifiable include files. Each JSP file in this directory defines HTML and script that might be used on more than one web page, and should not be modified.

- `domain_specific`

  Contains `selectCAdomain.jsp`, which an installation would modify if it had multiple domains with a single administrator.

- `WEB-INF`

  Contains supporting Java classes, including servlets, that an installation is not expected to modify, and `web.xml`, which an installation needs to modify to add additional applications or domains. (See "Creating application domains when you use JSPs to implement the web application" on page 301.)

Table 46. JSP files in the `Customers` directory

File name	**Description**
`pkimain.jsp`	Starting point for all tasks available to the end user, such as requesting a new certificate, picking up a previously requested certificate, renewing or revoking a previously issued browser certificate, or recovering a certificate whose key was generated by PKI Services
`certrequest.jsp`	Web page to request a certificate.
`certretrieve.jsp`	Web page to retrieve a certificate.
`genericbad.jsp`	Displays error information when processing fails.
`genericok.jsp`	Notifies the user of successful processing when no additional information (such as a transaction ID) needs to be returned to the user.
`installcert.jsp`	Contains HTML and JavaScript, functions to install an automatically renewed certificate that is copied from an email notification if using the Internet Explorer browser.
`keygenrequestok.jsp`	Notifies the user of a successful certificate request with PKI-generated keys, and includes the email address where notification has been sent.
`pkcs10retrieved.jsp`	Returns a retrieved PKCS #10 certificate.
`qrecover.jsp`	Implements the web page to recover a certificate.

Table 46. JSP files in the `Customers` directory (continued)	
File name	**Description**
`recoverbad.jsp`	Displays information for an unsuccessful attempt to find certificates for which the private key is to be recovered.
`recoverok.jsp`	Displays results of a successful attempt to find certificates for which the private key is to be recovered.
`renew_revoke.jsp`	Displays a client-authenticated browser certificate for the user to renew, revoke, or suspend.
`requestok.jsp`	Returns results (such as the transaction ID) of a successful certificate request.
`requestbad.jsp`	Returns error information for an unsuccessful certificate request.
`retrievebad.jsp`	Returns error information for an unsuccessful attempt to retrieve a certificate.

Table 47. JSP files in the `PKIServ` directory	
File name	**Description**
`actcertok.jsp`	Returns results of a successful certificate action, such as deleting, revoking, suspending, or resuming a certificate.
`actcertbad.jsp`	Returns error information for an unsuccessful certificate action, such as deleting, revoking, suspending, or resuming a certificate.
`actrequestok.jsp`	Returns results of a successful action on a certificate request, such as approving, rejecting, or deleting.
`acttrequestbad.jsp`	Returns error information for an unsuccessful action on a certificate request, such as approving, rejecting, or deleting.
`adminmain.jsp`	The main PKI Services administration page, from which the administrator can search certificates or certificate requests, or enter a specific transaction ID or serial number to act upon.
`admhome.jsp`	The PKI Services administration home page, which contains links to the end-user home page for each application and to the main PKI Services administration web page (`admmain.jsp`)
`approvewithmods.jsp`	Displays the form for an administrator to approve a certificate request with modifications.
`certdetailsbad.jsp`	Displays error information for a failed attempt to display the details of a certificate.
`certdetailsok.jsp`	Displays details of a single certificate.
`genericfailure.jsp`	Displays error information when the requested administration processing fails.

Table 47. JSP files in the PKIServ directory (continued)

File name	Description
`modifybad.jsp`	Displays error information when an approve with modifications action fails.
`modifyok.jsp`	Displays results of successful approve with modification action.
`querycertok.jsp`	Displays search results when administrator queries certificates.
`queryreqok.jsp`	Displays search results when administrator queries certificate requests.
`reqdetailsok.jsp`	Displays details of a single certificate request.

Table 48. JSP files in the mod.inc directory

File name	Description
`adminbottomnav.jsp`	End of page navigation appearing on administrator's web pages
`adminfooter.jsp`	End footer (the last element) on administrator's web pages
`altdomain.jsp`	Form field for altdomain
`altemail.jsp`	Form field for altemail
`altipaddr.jsp`	Form field for altipaddr
`altother_1_2_3_4_5.jsp`	altother form field sample for OID 1.2.3.4.5
`altother_1_2_3_4_6.jsp`	altother form field sample for OID 1.2.3.4.6
`altother_1_3_6_1_4_1_311_20_2_3.jsp`	altother form field sample for OID 1.3.6.1.4.1.311.20.2.3
`alturi.jsp`	Form field for alturl
`autorenew.jsp`	Form field for autorenew
`bottomnav.jsp`	End of page navigation on end user web pages
`businesscat.jsp`	Form field for BusinessCat
`certrow.jsp`	Displays a row in a table about certificates, format tied to `PKIServ/certdetailsok.jsp`
`challengepassphrase.jsp`	Displays passphrase form field when user is prompted to match the passphrase previously entered
`clientname.jsp`	Form field for clientname
`commonfunctions.jsp`	Contains JavaScript functions that are commonly used, such as `trim()`
`commonname.jsp`	Form field for commonname
`country.jsp`	Form field for country
`CustomExt.jsp`	The HTML and JavaScript for defining custom certificate extensions.

Table 48. JSP files in the `mod.inc` directory (continued)

File name	Description
`date.jsp`	Displays validity period form fields as two dates, the date the certificate becomes valid, and the date the certificate expires, used on approve with modification processing
`dnqualifier.jsp`	Form field for distinguished name qualifier
`domainname.jsp`	Form field for domain name
`emailaddr.jsp`	Form field for emailaddr
`extkeyusage.jsp`	Form field for extended key usage
`footer.jsp`	Footer (last element) on end users pages
`hostidmap.jsp`	Form field for host id mapping
`jurcountry.jsp`	Form field for JurCountry
`jurlocality.jsp`	Form field for JurLocality
`jurstateprov.jsp`	Form field for JurStateProv
`keysize.jsp`	Form field for keysize
`keyusage.jsp`	Form field for key usage
`label.jsp`	Form field for label
`locality.jsp`	Form field for locality
`mail.jsp`	Form field for mail
`notafter.jsp`	Form field for the number of days the certificate should be valid (from the time of request)
`notbefore.jsp`	Form field for the number of days before the certificate should be valid
`notifyemail.jsp`	Form field for notifyemail
`org.jsp`	Form field for org
`orgunit.jsp`	Form field for orgunit
`passphrase.jsp`	Form field for user to enter passphrase
`postalcode.jsp`	Form field for postal code
`publickey.jsp`	"Public key" form field, text box for entering base64-encoded PKCS #10 certificate request
`publickey_smartcard.jsp`	The HTML and JavaScript for the public key form field, when the browser provides the public key after the user selects from a list of available cryptographic providers.
`recoveremail.jsp`	recoveremail form field
`requestor.jsp`	Requestor form field (with prompt for name for tracking this request)
`requestor2.jsp`	Requestor form field (with prompt for email address of requestor)

Table 48. JSP files in the `mod.inc` directory (continued)

File name	**Description**
`security.jsp`	Displays a form field for entering a response to a security question
`security2.jsp`	Displays a form field for entering a response to a second security question
`serialnumber.jsp`	Form field for entering a certificate's serial number
`signwith.jsp`	Form field for selecting the signer of the requested certificate
`stateprov.jsp`	Form field for stateprov
`street.jsp`	Form field for street
`title.jsp`	Form field for title
`transactionid.jsp`	Form field for entering a certificate request's transaction ID
`uid.jsp`	Form field for uid
`unstructaddr.jsp`	Form field for unstructaddr
`unstructname.jsp`	Form field for unstructname
`userid.jsp`	Form field for userid

Chapter 14. Advanced customization

This topic describes the advanced customization procedures available for PKI Services. All are optional.

- Chapter 14, "Advanced customization," on page 283 explains:
  - "Scaling for high volume installations" on page 283
  - "Using certificate policies" on page 284
  - "Updating the signature algorithm" on page 287
  - "Customizing distribution point CRLs" on page 290
  - "Creating a distribution point ARL" on page 295
  - "Enabling support for large CRLs" on page 297
  - "Using the OCSP responder" on page 298
  - "Adding an application domain" on page 298
  - "Adding a new CA domain" on page 302
  - "Customizing email notifications sent to users" on page 317
  - "Setting up automatic renewal of certificates" on page 323
  - "Setting up PKI Services to generate keys for certificate requests" on page 325
  - "Adding custom extensions to certificates" on page 328
- Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333
- Chapter 16, "Using Enrollment over Secure Transport (EST)," on page 341

Scaling for high volume installations

Some PKI Services installations manage many certificates and certificates requests. The following guidelines can help you scale your system to maintain high performance in a high volume environment.

Guidelines:

1. Use distribution point CRLs if you average more than 500 revoked non-expired certificates at any given time. For more information, see "Customizing distribution point CRLs" on page 290.
2. If you anticipate having many certificate requests pending approval at any given time, implement a PKI exit to automate the approval process. (For more information, see Chapter 17, "Customizing with installation exit routines," on page 349.) This need arises from the human limitation rather than a technical one because it becomes nearly impossible to manually approve the requests when the volume grows too high.
3. To prevent name collisions in the LDAP directory, ensure that the subject distinguished names are unique. This can either be done by implementing a PKI exit to supply a unique name, or by enforcing the use of the `MAIL=` distinguished name attribute where you require the email address to be unique.
4. Queries against the request or ICL database can time out if the database contains many records. The performance of the query can be vastly improved by supplying the requester's name as additional search criteria if the saved requester data is meaningful to your organization and it is recallable. In this case, a PKI exit can be used to supply a meaningful value, such as a Lotus® Notes® short name or customer account number.
5. Keep the size of the request and ICL databases small by quickly removing records that are no longer needed. This can be done by setting low values for the following fields in the **ObjectStore** section of the PKI Services configuration file (`pkiserv.conf`):
   - `RemoveCompletedReqs`
   - `RemoveInactiveReqs`

- RemoveExpiredCerts

Using certificate policies

Certificates can contain a CertificatePolicies extension. This extension contains policy information, such as how your CA operates and the intended purpose of the issued certificates. (For more information about this extension, see RFC 5280 (tools.ietf.org/html/rfc5280).)

The CertificatePolicies extension contains one or more PolicyInformation sequences. (Typical usage has just one of these.) The PolicyInformation sequence has the following format:

- Your Policy OID as registered with the appropriate standards organization (ISO or ITU)
- Zero or more PolicyQualifiers sequences, each having the following information:
  - Either a Certificate Practices Statement (CPS) URI
  - Or a UserNotice sequence, which consists of one or both of the following text strings:
    - A notice that is intended to be viewed by customers using the certificate such as copyright or other legal information
    - Your organization's legal name with one or more notice numbers defined elsewhere, perhaps in your CPS.

By default, PKI Services does not include this extension in the certificates it creates. However, you can define your own CertificatePolicies extension by modifying fields in the **CertPolicy** section of the `pkiserv.conf` configuration file. You can also specify the `PolicyRequired` value to indicate whether a CertificatePolicies extension should be created for all certificate templates on a global basis or whether one is individually created based on the specifications of each certificate template.

PolicyRequired=T
Indicates that the CertificatePolicies extension is added to all certificates, and includes all `PolicyName`*n* values specified in the `pkiserv.conf` file. Policies that are specified in the CertPolicies input parameter or listed in the CONSTANT section of the template used to generate the certificate are ignored.

See "Steps for creating the CertificatePolicies extension on a global basis" on page 284.

PolicyRequired=F (default)
Indicates that the CertificatePolicies extension is added to certificates only when a certificate policy is specified in the CertPolicies input parameter or in the template when a certificate is requested. If you are implementing the web application using REXX CGI execs, the specification is done in the CONSTANT section of `pkiserv.tmpl`. If you are using JavaServer pages (JSPs), the specification is done with the CertPolicies tag in `pkitmpl.xml`.

See "Steps for creating the CertificatePolicies extension on a template basis" on page 286.

Note: PolicyCritical is ignored unless `PolicyRequired=T`. When `PolicyRequired=F`, setting `%%Critical=CertPolicies%%` in the CONSTANT section of the template marks the extension critical.

Restriction: When policies are specified within an individual template, the policy data is saved with the request at the time the request is submitted or modified. Therefore, if PKI Services is stopped and restarted to make changes in the policy data before the certificate is issued, the changes are not reflected in the issued certificate. However, the `PolicyRequired=F` setting is checked at the time the certificate is issued. Therefore, if PKI Services is stopped and restarted to make changes to the `PolicyRequired` setting before the certificate is issued, the new setting is used to determine which policy information is used (the global policy data or the data saved with the request.)

Steps for creating the CertificatePolicies extension on a global basis

Perform the following steps to create your own CertificatePolicies extension on a global basis:

1. Edit the `pkiserv.conf` configuration file and find the **CertPolicy** section.

____

2. Change the value of `PolicyRequired` to T (True) as in the following line:

   ```
   PolicyRequired=T
   ```

____

3. If you want to have the extension marked critical (this is not suggested), set the `PolicyCritical` equal to T (True) as in the following line:

   ```
   PolicyCritical=T
   ```

____

4. Go to the **OIDs** section of the `pkiserv.conf` configuration file. By default (as shown in the following example), the name is `MyPolicy=1.2.3.4` and value is `1.2.3.4`. The value of MyPolicy should be an installation-specific (registered) Object ID identifying your organization's certificate. Replace the value of MyPolicy in the following line with your Object ID.

   Example:

   ```
   [OIDs]
   MyPolicy=1.2.3.4
   ```

   Optionally, change the parameter name `MyPolicy` to your own installation-specific name. If you change the parameter name in this step, make a note of it. You need it for the next step. You can repeat the `MyPolicy` parameter using unique names and values if you need to define multiple policies.

   Example:

   ```
   MyPolicy=1.2.3.4
   MyOtherPolicy=2.3.4.5
   ```

____

5. If you changed the parameter name `MyPolicy` in the previous step, go back to the **CertPolicy** section and update the `PolicyName1` line to change the `MyPolicy` parameter to the policy name you specified in the **OIDs** section:

   ```
   [CertPolicy]
   PolicyName1=MyPolicy
   ```

____

6. If you want to add qualifiers, perform the following steps:

   a. Uncomment the following lines by removing the "#" characters and update the Policy1Org and Policy1Notice*n* fields:

      ```
      #Policy1Org=MyOrganization
      #Policy1Notice1=3
      #Policy1Notice2=17
      ```

      Policy1Org
         Your organization's name, for example, `International Business Machines, Inc.`

      Policy1Notice1 through Policy1Notice*n*
         Your notice numbers. (You might need more than one Policy1Notice*n* line, depending on how many notice numbers you have. Repeat the line as needed, by incrementing the suffix number on the keyword, for example Policy1Notice1, Policy1Notice2, and so forth.)

   b. Change the value of the UserNoticeText1 line shown in the following sample. The *statement* should be your notice text string, for example, `Certificate for IBM internal use only`. It cannot be longer than 200 characters, and must not contain embedded control characters (such as tab, carriage return, and line feed).

```
UserNoticeText1=statement
```

Note: Starting in z/OS V2R1, PKI Services encodes the UserNoticeText1 data as a UTF8String. In earlier releases it was encoded as a VisibleString, which is not allowed by RFC 5280 (tools.ietf.org/html/rfc5280).

c. Change the value of the CPS1 line shown in the following sample. The value should be your CPS URI, for example, `http://www.ibm.com/cps.html`.

```
CPS1=http://www.mycompany.com/cps.html
```

If you do not want to add qualifiers, delete or comment out (by inserting a # character at the start of the line) the preceding lines.

____

7. If you need multiple qualifiers, repeat the following fields as needed, incrementing the suffix numbers, for example:

```
PolicyName2=MyOtherPolicy
Policy2Org=International Business Machines, Inc.
Policy2Notice1=5
Policy2Notice2=9
UserNoticeText2=Certificate is intended for testing only
CPS2=http://www.ibm.com/cps2.html
```

____

8. If you made any changes to the PKI Services configuration, stop, and restart PKI Services to activate the changes.

____

Steps for creating the CertificatePolicies extension on a template basis

Perform the following steps to create your own CertificatePolicies extension on an individual template basis:

1. Edit the `pkiserv.conf` configuration file and find the **CertPolicy** section.

____

2. Change the value of `PolicyRequired` to F (False) as in the following line:

```
PolicyRequired=F
```

____

3. Follow steps "4" on page 285 through "7" on page 286 in "Steps for creating the CertificatePolicies extension on a global basis" on page 284 to create the individual policies you need.

____

4. Update the certificate template to specify the CertificatePolicies extensions that are to be created for it.

   - **If you are implementing the web application using REXX CGI execs:** Edit `pkiserv.tmpl` and customize the CONSTANT subsection under the certificate template for which you need CertificatePolicies extensions.

     For example, if you have specified values for PolicyName1, PolicyName3, and PolicyName6 in `pkiserv.conf`, then you can specify the certificate policies in `pkiserv.tmpl` in the following ways:

```
%%CertPolicies=3%%
 or
```

```
%%CertPolicies=3 6%%
 or
%%CertPolicies=1 3 6%%
```

If you want to make the CertPolicies extension critical, specify the following in the CONSTANT section:

```
%%Critical=CertPolicies%%
```

- **If you are implementing the web application using JavaServer pages (JSPs):** Edit `pkitmpl.xml` and customize the section for the certificate template for which you need CertificatePolicies extensions.

  For example, if you have specified values for PolicyName1, PolicyName3, and Place-name in `pkiserv.conf`, then you can specify the certificate policies in `pkitmpl.xml` in the following ways:

  ```
  <tns:CertPolicies>3</tns:CertPolicies>
   or
  <tns:CertPolicies>3 6</tns:CertPolicies>
  or
  <tns:CertPolicies>1 3 6</tns:CertPolicies>
  ```

  If you want to make the CertPolicies extension critical, specify the following tag in the certificate template section:

  ```
  <tns:Critical>CertPolicies</tns:Critical>
  ```

Rule: The policy numbers in the template file must exist in the `pkiserv.conf` file. For each template, you can choose a different subset of these numbers.

____

5. If you made any changes to the PKI Services configuration, stop and restart PKI Services to activate the changes.

____

Updating the signature algorithm

The signature algorithm that PKI Services uses to sign certificates must be based on the key type of the CA certificate. If it is not, PKI Services is unable to start. By default, IKYSETUP creates the CA certificate with an RSA key pair. The default value of the signature algorithm in the `pkiserv.conf` file is `sha-256WithRSAEncryption`. You can change the signature algorithm by changing the `SigAlg1` value in the **CertPolicy** section of the `pkiserv.conf` configuration file. Set `SigAlg1` to one of the algorithm identifiers shown in Table 49 on page 287 for the key type of the CA certificate.

Table 49. Supported signature algorithms for each CA certificate key type

CA certificate key type	Algorithm nickname	Algorithm OID	Applicable FIPS mode
RSA	sha-1WithRSAEncryption	1.2.840.113549.1.1.5	0 and 1
	sha-224WithRSAEncryption	1.2.840.113549.1.1.14	0, 1, 2, and 3
	sha-256WithRSAEncryption	1.2.840.113549.1.1.11	0, 1, 2, and 3
	sha-384WithRSAEncryption	1.2.840.113549.1.1.12	0, 1, 2, and 3
	sha-512WithRSAEncryption	1.2.840.113549.1.1.13	0, 1, 2, and 3
DSA	id-dsa-with-sha1	1.2.840.10040.4.3	0 and 1
	id-dsa-with-sha224	2.16.840.1.101.3.4.3.1	0, 1, 2, and 3

Table 49. Supported signature algorithms for each CA certificate key type (continued)

CA certificate key type	Algorithm nickname	Algorithm OID	Applicable FIPS mode
	`id-dsa-with-sha256`	`2.16.840.1.101.3.4.3.2`	0, 1, 2, and 3
`ECC`	`ecdsa-with-sha1`	`1.2.840.10045.4.1`	0 and 1
	`ecdsa-with-sha224`	`1.2.840.10045.4.3.1`	0, 1, 2, and 3
	`ecdsa-with-sha256`	`1.2.840.10045.4.3.2`	0, 1, 2, and 3
	`ecdsa-with-sha384`	`1.2.840.10045.4.3.3`	0, 1, 2, and 3
	`ecdsa-with-sha512`	`1.2.840.10045.4.3.4`	0, 1, 2, and 3
`RSASSA-PSS`	`sha-256Hash`	`2.16.840.1.101.3.4.2.1`	0, 1, 2, and 3
	`sha-384Hash`	`2.16.840.1.101.3.4.2`	0, 1, 2, and 3
	`sha-512Hash`	`2.16.840.1.101.3.4.2.3`	0, 1, 2, and 3

Tips: Consider these points when choosing the signature algorithm:

- sha-256WithRSAEncryption is more secure than sha-1WithRSAEncryption, but some browsers cannot install certificates that use sha-256WithRSAEncryption.
- The National Institute of Standards and Technology (NIST) recommends that SHA1 no longer be used.

Steps for changing the signature algorithm

Before you begin

Change the signing algorithm *before* you create any certificate requests. If changing the signing algorithm *after* some certificates requests have been created, you must wait until all requests are approved and the certificates created, or else you must add a `SigAlg2=`*`old-signing-algorithm`* line to the **CertPolicy** section. If you take this second option, `SigAlg1` becomes the signature algorithm for new requests.

Procedure

Perform the following steps to change the signature algorithm:

1. Edit the `pkiserv.conf` configuration file and find the **OIDs** section.

   ____

2. Ensure that the OID for the signature algorithm that you want to use is listed. (If you are using a version of `pkiserv.conf` from a previous release, OIDs added in later releases are missing if you have not added them.) If the OID is missing, add it by adding a line with the following format:

   ```
   algorithm_nickname=algorithm_OID
   ```

   where the values of *algorithm_nickname* and the corresponding *algorithm_OID* are from Table 49 on page 287. For example, if you want to use SHA512,

   ```
   sha-512WithRSAEncryption=1.2.840.113549.1.1.13
   ```

   ____

3. Find the **CertPolicy** section. If you want to use a signature algorithm other than SHA256, change `sha-256WithRSAEncryption` in the following line to the value for the signature algorithm that you want to use. The supported values for each CA certificate key type are shown in Table 49 on page 287.

   ```
   SigAlg1=sha-256WithRSAEncryption
   ```

   Note: If your configuration file is from a release before z/OS V1R12 and contains the default value, the line for the signature algorithm is:

   ```
   SigAlg1=sha-1WithRSAEncryption
   ```

   ____

Certificate revocation status

PKI Services provides the following for certificate revocation status checking:

- Certificate revocation lists.

  Forms of revocation lists supported are:

  – The global certificate revocation list (global CRL) which represents all of the non-CA certificates that are created by an instance of PKI Services.
  – The global authority revocation list (global ARL) which represents all of the subordinate CA certificates that are created by an instance of PKI Services.
  – The distribution point certificate revocation lists (DP CRLs) which represent a subset of the non-CA certificates that are created by an instance of PKI Services based on the serial number of the issued certificate.
  – The distribution point authority revocation list (DP ARL) which represents all the subordinate CA certificates that are created by an instance of PKI Services.
- Online Certificate Status Protocol (OCSP).

How distribution point CRLs/ARLs work

PKI Services always creates a *global* CRL regardless of whether you choose to use DP CRLs. The global CRL contains revocation information for certificates that have no CRLDistributionPoints extension (in other words, certificates defined with `CRLDistSize=0`). When a certificate contains a CRLDistributionPoints extension, PKI Services publishes its revocation status to the appropriate DP CRL, not in the global CRL.

The following topics help you understand more about how DP CRLs work. This information is useful if you write applications that process CRLs.

How DP CRLs are published

DP CRLs are published to LDAP at leaf nodes directly following the CA's entry. For example, if the CA's name is:

```
OU=My Company Certificate Authority,O=My Company,C=US
```

Then, the DP CRLs would be published to:

```
CN=DP-name,OU=My Company Certificate Authority,O=My Company,C=US
```

How DP CRLs are partitioned

The partitioning of the overall CRL into partial CRLs is based on certificate serial number and the value of CRLDistSize in `pkiserv.conf`. For example, if CRLDistSize is `100` and CRLDistName is ABC, then certificates with serial numbers `1–100` appear on DP `ABC1`; `101–200` on DP `ABC2`, and so on. PKI Services dynamically creates DP CRLs as needed as a part of certificate issuance. Existing DP CRLs are refreshed along with the global CRL during CRL interval processing.

As certificates expire, they are no longer eligible for revocation and do not appear on any CRL. Therefore, over time, each distribution point becomes inactive. PKI Services automatically retires DP CRLs that become inactive by no longer publishing their CRLs. However, retired DP CRLs previously published to LDAP remain in LDAP. PKI Services makes no attempt to delete these.

Even when using distribution point CRLs, the single non-DP CRL (global CRL) is still created. Revoked certificates containing the CRLDistributionPoints extension appears only on the appropriate DP CRL, not the global CRL.

What about CA certificates?

PKI Services can be used to create other subordinate certificate authority certificates. Since revocation activity against these CA certificates is normally low, PKI Services, by default, does not partition authority revocation lists (ARLs). You can choose to create a distribution point ARL in a single partition for checking the revocation status of CA certificates. (See "Creating a distribution point ARL" on page 295.) When you choose to create a DP ARL, your CA certificates contain a CRLDistributionPoints extension.

When you do not choose to create a DP ARL (`ARLDist=F`), applications wanting to check the revocation status of a CA certificate must check the global ARL. In addition, when `ARLDist=F`, CA certificates do not contain a CRLDistributionPoints extension, although they are treated as if they had the extension when determining the partitioning of the global CRL. For instance, with `ARLDist=F` and `CRLDistSize=10`, if you issue 10 CA certificates plus one non-CA certificate, the non-CA certificate information would be published to the second distribution point CRL. (The first DP CRL would remain empty.)

Customizing distribution point CRLs

If your PKI Services installation is very active, many certificates can be in the revoked state at any one time. Therefore, the certificate revocation list (CRL) can become quite large, causing considerable network traffic and overhead to an application wanting to process it. Publishing partial CRLs to multiple distribution point (DP) CRLs is a way of keeping your CRLs small.

Guideline: Consider using distribution point CRLs if you anticipate averaging more than 500 revoked non-expired certificates at any given time.

You begin using distribution point CRLs when you accept the defaults settings contained in PKI Services configuration file (`pkiserv.conf`). You can customize those settings by specifying the number of certificates per DP CRL and by specifying the name of the DP CRL using the following two parameters in the **CertPolicy** section of the `pkiserv.conf`:

`CRLDistSize`
Specifies the maximum number of certificates to be managed by a single DP. This represents the number of entries in each DP CRL if all active certificates are revoked at once.

`CRLDistName`
Specifies the file name, or the constant portion of the leaf-node RDN, for the CRL distribution point.

You can choose to further customize your DP CRL processing to build the URI format name for the distribution point in the CRLDistributionPoints extension of each certificate. This allows your certificate validation programs to dynamically retrieve a CRL without being preconfigured with LDAP bind information. However, because bind credentials cannot be added to DP CRLs with URI format names, anonymous access is used to retrieve the CRL.

The URI format name is built in *addition* to the LDAP distinguished name of the DP CRL that is always added when `CRLDistSize` is greater than zero. You can add the URI format name by customizing the following two parameters in the **CertPolicy** section of the `pkiserv.conf`:

`CRLDistURI`*`n`*
Specifies the name for the DP CRL in the form of a URI that adds the protocol type and the server domain name.

`CRLDistDirPath`
Specifies the full path for the file system directory where PKI Services saves each DP CRL.

You can also choose to have PKI Services create a CRLDistributionPoints extension for each CA certificate in addition to non-CA certificates. You choose this by customizing the `ARLDist` parameter in the **CertPolicy** section of the `pkiserv.conf`. This creates a distribution-point authority-revocation list (DP ARL) for your CA certificates. See "Creating a distribution point ARL" on page 295 for details.

Specifying the URI format

When you choose to use distribution points for CRL and ARL processing, PKI Services updates the CRLDistributionPoints extension with the distinguished name for the LDAP entry where the distribution point is posted. You can choose to add another name to the extension in the URI format which contains the protocol type and the server domain name in addition to the distinguished name. With the URI format, the location of the distribution point is self-contained in the CRLDistributionPoints extension.

The URI format contains the following information:

- The protocol type (LDAP or HTTP).
- The server domain name.
- If the protocol is LDAP:
  - The distinguished name of the distribution point.
  - For non-CA certificates, the attribute string `?certificateRevocationList`.
  - For CA certificates, the attribute string `?authorityRevocationList`.
- If the protocol is HTTP, the virtual or real path name, ending with the file name - formed from the common name portion of the distinguished name of the distribution point with the `.crl` extension - where the distribution point CRL is stored.

Examples:

```
ldap://ldap.bankxyz.com:389/CN=CRLlist1,OU=Bank XYZ
       Authority,O=Bank XYZ,C=US?certificateRevocationList

http://www.bankxyz.com/PKIServ/cacerts/CRLlist1.crl
```

Note: This is an example of an HTTP protocol URI using a virtual path name. When using virtual path names in an HTTP URI, a Pass statement is required in the HTTP configuration file to map the virtual path name to a real path name. See "Determining CRLDistDirPath" on page 293 for additional information.

Restriction: Special characters, such as spaces, quotation marks, and square brackets are not considered *safe* to use in URLs and should be encoded using the appropriate *escape* sequence. For details, see RFC 1738 (tools.ietf.org/html/rfc1738).

Determining CRLDistURI*n*

If you are using DP CRLs (you specified a `CRLDistSize` value greater than **1** in the **CertPolicy** section of `pkiserv.conf`), you can choose to further customize your DP CRL processing to build the URI format name for the DP CRL in the CRLDistributionPoints extension of each certificate. The URI format name is built in *addition* to the LDAP distinguished name of the DP CRL, as described in "Specifying the URI format" on page 291.

This is an optional parameter. If you do not specify a `CRLDistURI`*`n`* value, the URI format name is not created. You can specify multiple entries for the `CRLDistURI`*`n`* parameter, using the parameters `CRLDistURI1`, `CRLDistURI2`, and so forth. This value is ignored if you did not specify `CRLDistSize` with a value greater than zero. The URI format is not created if you specify `CRLDistURI`*`n`* with an *n* value of **0**.

There are different ways to specify the value of `CRLDistURI`*n* for different protocols. **Valid values** include *one* of the following strings:

- A string that begins with the characters `http://` or `ldap://`
- A string that consists of `LdapServer`*n*, where *n* is greater than zero.

Restriction: PKI Services provides syntax checking based only on valid values for the `CRLDistURI`*n* value. You must ensure that the URIs you choose can be accessed.

Specifying an HTTP URI

For HTTP, specify the complete URL but do not specify the name of a file where the CRL DP is stored. The value for `CRLDistURI`*n* can be specified with or without a trailing slash.

Example:

```
CRLDistURI1=http://www.bankxyz.com/PKIServ/cacerts/
```

Note: This is an example of an HTTP protocol URI using a virtual path name. When using virtual path names in an HTTP URI, an AliasMatch HTTP directive is required in the `vhost80.conf` (host file for son-SSL requests) configuration file to map the virtual path name to a real path name. See "Determining CRLDistDirPath" on page 293 for more information.

Specifying an LDAP URI

For LDAP, there are two ways to indicate the `CRLDistURI`*n* value. Choose either of the following two methods:

- Specify the protocol and the domain name (and the port, if needed). The value for `CRLDistURI`*n* can be specified with or without a trailing slash.

  Example:

  ```
  CRLDistURI1=ldap://ldap.bankxyz.com:389/
  ```

- Specify the keyword `LdapServer`*n* to have PKI Services build the `CRLDistURI`*n* value for you based on a server identified by the `Server`*n* or `BindProfile`*n* directives in the **LDAP** section of `pkiserv.conf`.

  Example:

  ```
  CRLDistURI3=LdapServer1
  ```

  This example assumes that the first server specified in the **LDAP** section was similarly defined as one of the following examples:

  Examples:

  ```
  Server1=ldap.bankxyz.com:389
            or
  BindProfile1=LOCALPKI.BINDINFO.LDAP1
  ```

  Rules for using the `LdapServer`*n* keyword:

  1. You must have specified a value greater than zero for `NumServers` in the **LDAP** section of `pkiserv.conf`.
  2. Each server represented by the *n* value in the `LdapServer`*n* keyword must be identified in *one* of the following ways:
     - The corresponding LDAP server must be identified by a `Server`*n* or `BindProfile`*n* value in the **LDAP** section of `pkiserv.conf`, *or*
     - The corresponding LDAP server must be identified in the default FACILITY class profile IRR.PROXY.DEFAULTS and must follow the same identification requirements for PKI Services LDAP processing. See "Using encrypted passwords for LDAP servers" on page 501.

Determining CRLDistDirPath

If the protocol for the URI you specified with `CRLDistURI`*n* is HTTP protocol, you need to also determine your value for the `CRLDistDirPath` parameter. The `CRLDistDirPath` parameter specifies the full path of the `var` directory where PKI Services saves each DP CRL. The value can be specified with or without the trailing slash. The default value is `/var/pkiserv/`. If you are customizing this value for a CA Domain, you should specify a directory name that contains the CA Domain name, for example `/var/pkiserv/employees/`. In a case such as this, it is also necessary to add an additional Pass statement to the HTTP configuration file that maps the virtual path name in the URI to the real path specified by this CRLDistDirPath statement.

Statements in the `pkiserv.conf` file:

```
CRLDistURI1 = http://www.bankxyz.com/Employees/cacerts/
CRLDistDirPath = /var/pkiserv/employees/
```

Matching AliasMatch HTTP directive in the `vhost80.conf` (host file for non-SSL request) configuration file.

Uncomment the AliasMatch HTTP directive in the `vhost80.conf` (host file for non-SSL request) configuration file.

```
#AliasMatch /Employees/cacerts/(.*) /var/pkiserv/employees/$1
```

The default value is `/var/pkiserv/`. See "Specifying the URI format" on page 291. This value is ignored if you do not create a CRLDistributionPoints extension or if the URI protocol is LDAP.

Steps for customizing distribution point CRLs

Before you begin

Be aware of the following **restrictions**:

- If running PKI Services in a sysplex, all instances of PKI Services must specify the same values for the parameters `CRLDistURI`*n* (for all values of *n*), `CRLDistDirPath`, `CRLDistSize`, and `CRLDistName`.
- Once a value for `CRLDistName` has been set, it must not be changed or removed from the configuration file.
- Once a nonzero value has been set for `CRLDistSize`, it must not be changed back to zero or removed from the configuration file. Adjusting the value is acceptable.

Procedure

Perform the following steps to customize distribution point CRLs:

1. Determine your value for the `CRLDistSize` parameter based on the following algorithm. The default value specified in `pkiserv.conf` is `500`. Your value should be based on your wanted average number of CRL entries per distribution point and your estimated revoked-certificate percentage as expressed by the following formula:

   ```
   CRLDistSize = E / P
   ```

   where:

   E
   is the wanted average number of CRL entries per distribution point.

   P
   is the estimated revoked-certificate percentage.

   Example: If you estimate that 10% of the non-expired certificates are in the revoked state at any time and you want the CRLs to average about 100 entries each, then:

   ```
   CRLDistSize = 100 / 0.10 = 1000
   ```

The `CRLDistSize` in bytes can be roughly estimated to be 500 + (25 × *number of CRL entries*). Using the previous example, the average CRL size in bytes would be 500 + (25 × 100) = 3000 bytes.

Note: The longer the CRL, the longer it takes to process it.

Restriction: When CRLs are posted to LDAP, a single CRL cannot exceed approximately 32 K bytes in length, unless you have enabled support for large CRLs. For more information, see "Enabling support for large CRLs" on page 297. Therefore, if a CRL is posted to LDAP and you have not enabled support for large CRLs, you must limit the length of the CRL.

Rules:

a. The value of `CRLDistSize` is a numeric value from 0 - 2147483647.
b. A nonzero value indicates that distribution point (DP) CRLs are created.
c. A value of zero (the default) indicates that DP CRL processing is not performed.

Guideline: If you anticipate a low revocation rate for active certificates, use a value of **0**. Your installation might not need to use distribution point CRLs and the global CRL might be sufficient.

____

2. If necessary, update the value of `CRLDistSize` in the **CertPolicy** section of `pkiserv.conf` to the customized value you determined in Step "1" on page 293.

   If you selected the **0** value for `CRLDistSize`, complete this step and then continue with Step "12" on page 295.

____

3. Determine your value for the `CRLDistName` parameter. The default value is `CRL`. The common name portion of the distinguished name of each DP CRL is formed by appending the DP number to this value. The CA's name is also appended. (See "How DP CRLs are published" on page 289.)

   Example:

   ```
   CN=CRL3,OU=My Company Certificate Authority,O=My Company,C=US
   ```

   Restrictions:

   a. The value of `CRLDistName` must contain only alphanumeric characters.
   b. The length of the entire DP distinguished name should not exceed 255 bytes. (DP distinguished names that are longer appear truncated in the PKIDPUBR audit record.)

____

4. If necessary, update the value of `CRLDistName` in the **CertPolicy** section of `pkiserv.conf` to your customized value.

____

5. Optionally, determine your value for the `CRLDistURI`*n* parameter. Specifying this value allows PKI Services to build a URI-formatted name for the DP CRL in each CRLDistributionPoints extension, if you also specified a `CRLDistSize` value greater than **1** in Step "2" on page 294. The URI format name is built in *addition* to the LDAP distinguished name of the DP CRL in each CRLDistributionPoints extension. If you do not specify a `CRLDistURI` value, the URI format name is not created. See "Specifying the URI format" on page 291.

   You can specify multiple entries for the `CRLDistURI`*n* parameter, using the parameters `CRLDistURI1`, `CRLDistURI2`, and so forth.

____

6. If necessary, update the value of `CRLDistURI`*n* in the **CertPolicy** section of `pkiserv.conf` to your customized value or values.

7. If all the protocol definitions for the URIs you specified with `CRLDistURI`*n* in Step “5” on page 294 are the LDAP protocol, decide whether you want to enable support for large CRLs. If you choose to enable large CRLs, follow the instructions in “Enabling support for large CRLs” on page 297. Then skip to Step “10” on page 295.

8. If a protocol definition for the URI you specified with `CRLDistURI`*n* in Step “5” on page 294 is HTTP protocol, determine your value for the `CRLDistDirPath` parameter.

   The `CRLDistDirPath` parameter specifies the full path of the `var` directory where PKI Services saves each DP CRL. The default value is `/var/pkiserv/`. The value can be specified with or without the trailing slash. See “Determining CRLDistDirPath” on page 293.

9. If necessary, update the value of `CRLDistDirPath` in the **CertPolicy** section of `pkiserv.conf` to your customized value.

10. Optionally, determine your value for the `ARLDist` parameter. Specifying this parameter creates a distribution point ARL so you can check revocation status for CA certificates without accessing the global ARL. See “Creating a distribution point ARL” on page 295.

11. If necessary, update the value of `ARLDist` in the **CertPolicy** section of `pkiserv.conf` to your customized value.

12. If you made any updates to `pkiserv.conf`, stop and restart PKI Services to make your changes effective.

When you have finished: If you selected a `CRLDistSize` value greater than zero, you have set up distribution point CRLs. Now, created certificates contain the CRLDistributionPoints extension indicating the location of the DP CRL that is checked for revocation information. If you specified a URI-formatted name with `CRLDistURI`*n*, now your CRLDistributionPoints extensions also contains a URI name for each DP CRL, containing the protocol type and server domain name. If you enabled the `ARLDist` option, you have set up a distribution point ARL for CA certificates.

Creating a distribution point ARL

You can choose to create a distribution point (DP) authority revocation list (ARL) to support revocation status checking for certificate authority (CA) certificates. You choose DP ARL processing by customizing the `ARLDist` parameter in the **CertPolicy** section of the `pkiserv.conf`. If you do not customize this parameter, PKI Services does not partition the ARL and, therefore, applications must check the global ARL to check the revocation status of a CA certificate.

ARLDist=F (default)
No distribution point ARL is created.

ARLDist=T
When distribution point CRLs are also enabled (when `CRLDistSize` is greater than zero), you can specify **T (True)** to create a distribution point ARL.

When DP ARL processing is enabled, PKI Services provides the following support:

- Create a single distribution point (DP) for all CA certificates
- Build a CRLDistributionPoints extension containing both the distinguished name and the URI format for the DP. Use the same values specified (`CRLDistSize, CRLDistName, CRLDistURIn, CRLDistDirPath`) in the `pkiserv.conf` file for the DP CRL processing.

DP ARL processing for CA certificates is similar to the DP CRL processing for non-CA certificates with the following differences:

- There is only one DP ARL. Its name is formed by the value that is specified in the `CRLDistName` parameter in the **CertPolicy** section of the `pkiserv.conf`, appended with **0** (zero). By appending a zero, the name of the DP ARL never conflicts with the name of a DP CRL. For example, if `CRLDistName=CRL`, then the DP ARL is named `CRL0`, and the DP CRLs are named `CRL1`, `CRL2`, and so forth.
- The DP ARL is a mirror copy of the global ARL. In other words, each revoked CA certificate appears in both the DP ARL and the global ARL. By contrast, a revoked non-CA certificate is listed in the DP CRL but not in the global CRL when DP CRL processing is enabled.
- The attribute string that is appended to the URI format for the LDAP protocol is ?`authorityRevocationList`. Otherwise, the CRLDistributionPoints extension of a CA certificate appears similar to that of a non-CA certificate. See Figure 33 on page 297 for a sample CRLDistributionPoints extension for a CA certificate. This sample contains several different name formats. Notice the URI format at the end of the sample.

```
SEQUENCE {
   OBJECT IDENTIFIER cRLDistributionPoints (2 5 29 31)
   OCTET STRING, encapsulates {
       SEQUENCE {
          SEQUENCE {
              [0] {
                [0] {
                  [4] {
                    SEQUENCE {
                      SET {
                        SEQUENCE {
                          OBJECT IDENTIFIER
                            countryName (2 5 4 6)
                          UTF8String (1997) 'US'
                          }
                      }
                      SET {
                        SEQUENCE {
                          OBJECT IDENTIFIER
                            organizationName (2 5 4 10)
                          UTF8String (1997) 'Mycompany'
                          }
                      }
                      SET {
                        SEQUENCE {
                          OBJECT IDENTIFIER
                            organizationalUnitName (2 5 4 11)
                          UTF8String (1997) 'Retail'
                          }
                      }
                      SET {
                        SEQUENCE {
                          OBJECT IDENTIFIER
                            commonName (2 5 4 3)
                          UTF8String (1997) 'CRL0'
                          }
                      }
                   }
                }
              }
          }
          SEQUENCE {
              [0] {
                [0] {
                  [6]
                  'http://crl.MyCompany.de/CRL0.crl'
                }
              }
          }
          SEQUENCE {
              [0] {
                [0] {
                  [6]

'ldap://ldap.MyCompany.de/CN=CRL0,OU=Retail,O=Mycompany,C=US?authorityRevocationList'
                }
              }
          }
       }
   }
 }
```

Figure 33. A sample CRLDistributionPoints extension for a certificate authority (CA) certificate

Enabling support for large CRLs

When LDAP posting of certificate revocation lists (CRLs) is enabled, by default PKI services temporarily stores CRLs in its object store for posting to LDAP. However, PKI Services imposes a limit on the size of records in the object store of approximately 32 KB, which limits the size of the CRLs stored there to approximately 32 KB. As certificates are revoked or suspended within the scope of a CRL, the size of the CRL increases, and can exceed the limit. If a CRL exceeds the 32 KB limit, PKI Services cannot post it to the LDAP directory.

To avoid this problem, you can configure PKI Services to store CRLs for posting to LDAP in the z/OS UNIX file system instead of in the object store. The distribution point CRLs and the distribution point ARLs are stored by using the same file name format that is used when an http format `CRLDistURIn` is specified. However, when large CRL support is enabled, the global CRLs and ARLs are also stored in the z/OS UNIX file system by using file names that are formed by using the `CRLDistName` value followed by `_MCRL.crl` or `_MARL.crl`. For example, if using `CRLDistName=CRL`, the global CRL file name is `CRL_MCRL.crl`. When you do this, there is no limit for the size of CRLs.

Steps for enabling support for large CRLs

Perform the following steps to enable support for CRLs larger than the limit of approximately 32KB.

Before you begin

You need to be familiar with the z/OS UNIX file systems.

Procedure

1. Configure a z/OS file system to hold CRLs. If you run multiple instances of PKI Services in a Parallel Sysplex (one per image), ensure that the file system is shared with each PKI Services instance in the sysplex. For information about managing the z/OS UNIX file system, see *z/OS UNIX System Services Planning*.

   ____

2. Set the value of the `LargeCRLPostPath` parameter in the **CertPolicy** section of the PKI Services configuration file, `pkiserv.conf`, to the full path of the `var` directory where PKI Services is to save each CRL for posting to LDAP. The default value is `/var/pkiserv/`. You can specify the value with or without the trailing slash. The value of `LargeCRLPostPath` can be the same as the value of `CRLDistDirPath`.

   Guideline: If you are customizing this value for a CA domain, specify a directory name that contains the CA domain name, for example `/var/pkiserv/employees/`, where `employees` is the domain name.

   ____

3. Set the value of the `EnableLargeCRLPosting` parameter in the **CertPolicy** section of the PKI Services configuration file, `pkiserv.conf`, to T.

   ____

4. Restart PKI Services. Your changes to the `pkiserv.conf` file do not take effect until you do this. For information about starting PKI Services see Chapter 10, "Starting and stopping PKI Services," on page 129.

   ____

Results

When you are done, you have enabled support for large CRLs, and CRLs are no longer limited to approximately 32KB in size.

Using the OCSP responder

As an alternative, or in addition to publishing revocation information with CRLs, you can choose to enable an Online Certificate Status Protocol (OCSP) responder. An OCSP responder is enabled when `OCSPType` is set to `basic` in the **CertPolicy** section of the PKI Services configuration file. See Table 21 on page 74, and the certificate contains the necessary OCSP responder information in the AuthInfoAccess extension. (Also, see "TEMPLATE sections" on page 151.)

To use an OCSP responder, you must add `/usr/lpp/pkiserv/lib` to the LIBPATH environment variable for the HTTP Server. This setting is shown by adding it to the `vhost80.conf` (host file for non-SSL requests) configuration file by using the SetEnv HTTP directive.

Adding an application domain

This topic describes adding a new application domain. If you want to add a new certificate authority (CA) domain, see "Adding a new CA domain" on page 302.

By default, all CGIs or JSPs (depending on the method you are using to implement the PKI Services web application) reside under a common URL. Based on this, all users, including PKI administrators, have the same PKI Services home page, web content and supported certificate templates. In other words, by default, there is a single application domain: PKISERV.

The sample PKI Services template files (`pkiserv.tmpl` and `pkitmpl.xml`) contain two application sections: PKISERV and CUSTOMERS. Using these two application sections, your users can be easily divided between two subsets - customers and administrators.

You might want to separate your administration users and your end users. You might also need to further subset your end user population by adding different application domains for different groups of end users. Both of these objectives can be accomplished by using multiple applications. The PKI administrators and the different subsets of end users can be directed to different application domains at different URLs.

Creating application domains when you use REXX CGIs to implement the web application

To create multiple application domains, execute the subtasks in Table 50 on page 299. Both are tasks for the web server programmer.

Table 50. Task roadmap for creating multiple application domains

Subtask	Associated procedures (See ...)
Update the PKI Services template file	"Steps for creating multiple application sections in the PKI Services template file" on page 299
Update the web server configuration files	"Steps for adding application domains to the web server configuration files" on page 300

Steps for creating multiple application sections in the PKI Services template file

Perform the following steps to create multiple application sections to the PKI Services template file:

1. Edit the `pkiserv.tmpl` file and find the CUSTOMERS application section.

   ____
2. Replicate the CUSTOMERS section and specify a unique name for the new application section.

   ```
   <APPLICATION NAME=appl-section-name>
   ```

   Rule: The application section name must be one word, all uppercase characters.

   ____
3. Determine which certificate types are required by this user subset. Based on these requirements, select the certificate templates that belong in the new application set by adding or removing template names from this new section as needed.

   ____
4. Customize the content of the web pages for this application by modifying the <CONTENT>...</CONTENT> subsection. (See "TEMPLATE sections" on page 151 for a description of each subsection.)

   ____
5. Similarly, customize the original CUSTOMERS application by re-executing Steps "3" on page 299 and "4" on page 299, this time editing the content of the CUSTOMERS web pages.

   ____
6. Repeat Steps "1" on page 299–"4" on page 299 for each application section you need to add.

   ____
7. Optionally, rename the original CUSTOMERS application to a new section name, if you want.

   ```
   <APPLICATION NAME=section-name>
   ```

Rule: The application section name must be one word, all uppercase characters.

____

Steps for adding application domains to the web server configuration files

Before you begin

- This procedure requires web server programming skills and requires editing the `vhost.conf` configuration file.
- The home page URL for the new or renamed domains would be as follows:

  ```
  https://<webserver-fully-qualified-domain-name>/<new-appl-domain-name>/ssl-cgi/camain.rexx
  ```

  where *new-appl-domain-name* corresponds to the new section name added in the template file in "Steps for creating multiple application sections in the PKI Services template file" on page 299. However, in the web server files, the new name is case-sensitive but does not need to be in uppercase only.
- Make note of the case you select for each character of the new *new-appl-domain-name* name. This case-sensitive value becomes part of the URL for your home page. You must use it consistently in each set of HTTP Server directives as indicated in the `vhost.conf` configuration files:

  `vhost80` - Virtual host file for non-SSL requests.
  `vhost443` - Virtual host file for SSL requests with server authentication.
  `vhost1443` - Virtual host file for SSL requests with client authentication.
- The administration home page URL does not change. (There is one common administration application that handles all application domains.)
- If your PKI installation has changed the name of the `Customers` domain, you must change all occurrences of `Customers` to its new value in both files. (The new value is not case-sensitive.)
- If your installation has added a new application domain, use the following procedure.

Procedure

Perform the following steps to add application domains or rename the Customers application domain in web server virtual host configuration files for each new application section added to `pkiserv.tmpl`:

- Modify each of the following HTTP Server directives in the virtual host configuration files:

  `vhost80.conf` (host file for non-SSL requests):

  ```
  RewriteRule ^/(PKIServ|Customers|Employees)/ssl-cgi/(.*) https://<server-domain-name>/$1/ssl-cgi-bin/$2 [R,NE]
  RewriteRule ^/(PKIServ|Customers|Employees)/clientauth-cgi/(.*) https://<server-domain-name>:1443/$1/clientauth-cgi-bin/$2
  [R,NE]
  ScriptAliasMatch /(PKIServ|Customers|Employees)/public-cgi/(.*) <application-root>/PKIServ/public-cgi/$2
  ```

  `vhost443.conf` (host file for SSL request with server authentication):

  ```
  RewriteRule ^/(PKIServ|Customers|Employees)/public-cgi/(.*) http://<server-domain-name>/$1/public-cgi/$2 [R,NE,L]
  RewriteRule ^/(PKIServ|Customers|Employees)/ssl-cgi/(.*) https://<server-domain-name>/$1/ssl-cgi-bin/$2 [R,NE]
  RewriteRule ^/(PKIServ|Customers|Employees)/clientauth-cgi/(.*) https://<server-domain-name>:1443/$1/ clientauth-cgi-bin/$2
  [R,NE,L]
  ScriptAliasMatch ^/(PKIServ|Customers|Employees)/(public-cgi|ssl-cgi-bin)/(.*) "<application-root>/PKIServ/$2/$3"
  ```

  ```
  <LocationMatch "^/(PKIServ|Customers|Employees)/ssl-cgi-bin(/(auth|surrogateauth))?/cagetcert.rexx">
  Charsetoptions TranslateAllMimeTypes
  </LocationMatch>
  ```

  `vhost1443.conf` (host file for SSL requests with client authentication):

  ```
  RewriteRule ^/(PKIServ|Customers|Employees)/public-cgi/(.*) http://<server-domain-name>/$1/public-cgi/$2 [R,NE,L]
  RewriteRule ^/(PKIServ|Customers|Employees)/ssl-cgi/(.*) https://<server-domain-name>/$1/ssl-cgi-bin/$2 [R,NE,L]
  ScriptAliasMatch ^/(PKIServ|Customers|Employees)/(clientauth-cgi|clientauth-cgi-bin)/(.*) "<application-root>/
   PKIServ/clientauth-cgi-bin/$3"

  <LocationMatch "^/(PKIServ|Customers|Employees)/clientauth-cgi-bin/auth/pkicmp">
  CharsetOptions NoTranslateRequestBodies
  </LocationMatch>
  ```

In contrast to the application section name, the domain name value is case-sensitive and does not need to be uppercase. However, you must use it consistently in each of the HTTP Server directives in the virtual host configuration files. This value becomes part of the URL for your home page.

When you are done, you have defined a new PKI Services application domain at:

Example

```
https://<webserver-fully-qualified-domain-name>/employees/ssl-cgi/camain.rexx
```

Creating application domains when you use JSPs to implement the web application

An application, or application domain, is a grouping of certificate request templates that are typically targeted at a subset of end users. The Customers application is the default end user application. But the `web.xml` file that ships as part of `PKIServ.ear` also defines mappings for a second application called Application2, making it easy to set up a second application domain.

To set up Application2, edit `pkitmpl.xml` and use the `<tns:application>` tag to define which certificate request templates should be part of Application2. For example:

```
<tns:application>
<tns:applname>Application2</tns:applname>
<tns:appltemplate>1-Year PKI SSL Browser Certificate</tns:appltemplate>
<tns:appltemplate>5-Year PKI SSL Server Certificate</tns:appltemplate>
<tns:appltemplate>5-Year PKI Intermediate CA Certificate</tns:appltemplate>
<tns:appltemplate>2-Year PKI Authenticode - Code Signing Certificate</
tns:appltemplate>
<tns:appltemplate>n-Year PKI Certificate for Extensions Demonstration</
tns:appltemplate>
<tns:appltemplate>1-Year SAF Browser Certificate</tns:appltemplate>
</tns:application>
```

While you access the Customers home page at the URL:

```
http://hostname.com:9080/PKIServ_Web/Customers/pkimain.jsp
```

you would access the Application2 home page at:

```
http://hostname.com:9080/PKIServ_Web/Application2/pkimain.jsp
```

To change the name of Application2, or to add another application, you need to edit the `web.xml` file in `PKIServ.ear`.

Steps for creating application domains other than Application2

Perform the following steps to create different application domains when you use JSPs to implement the web application.

Before you begin

You must have the **jar** command in your path. If you do not, define the JAVA_HOME variable (you can find its value on the WebSphere administration console) and add the `$JAVA_HOME/bin` directory to your path. For example:

```
export JAVA_HOME=/WebSphere/V6R1/AppServer/java
export PATH=$JAVA_HOME/bin:$PATH
```

Procedure

1. Follow steps "1" on page 259 to "4" on page 259.

   ____

2. Edit the `web.xml` file. You can use the following command, or use your preferred editor:

```
oedit WEB-INF/web.xml
```

3. Copy the lines that begin with the comment:

   ```
   <!-- Start: For new application: Application2 -->
   ```

   and end with the comment:

   ```
   <!-- End: For new application: Application2 -->
   ```

4. Change all occurrences of "Application2" in the copied lines to the name of the application you want to create. Save the updated file.

5. Update the WAR file with the edited template file.

   ```
   jar -uvf PKIServ_Web.war WEB-INF/web.xml
   ```

6. Update the EAR file with the updated WAR file:

   ```
   jar -uvf PKIServ.ear PKIServ_Web.war
   ```

7. Make sure the `PKIServ.ear` file is publicly readable by issuing the **chmod** command.

   ```
   chmod 755 PKIServ.ear
   ```

Results

When you are done, you created another application domain. You can now deploy the updated files following the directions in "Deploying the EAR file to a WebSphere Application Server" on page 261. When you edit the file `pkitmpl.xml` and add your new application, if its name is "NewApplication" your new application is available at:

```
http://hostname.com:9080/PKIServ_Web/NewApplication/pkimain.jsp
```

Adding a new CA domain

When you want to operate more than one certificate authority (CA) on a single z/OS image, you must create a separate CA domain for each CA. Each CA domain uses its own daemon and operates as its own instance of PKI Services. This topic describes how to add a new CA domain. (If you want to add a new *application* domain, see "Adding an application domain" on page 298.)

When you add CA domains, you can create a PKI infrastructure that contains subsets of end user populations (application domains), each supported by its own unique PKI Services application (PKI Services daemon and URL) and optionally by its own dedicated set of PKI administrators. If you already use multiple application domains, the key advantage of adding multiple CA domains is that you can build a certificate hierarchy of CAs and optionally provide certificate services to multiple organizations.

When you add a new CA domain, your users still have a unique URL and set of certificate templates to choose from, but they also have the services of their own CA including the CA's certificate, signing key, object store, and issued certificate list (ICL), and LDAP repository. Enabling multiple CAs is a natural extension for multiple application domains. Each CA domain can represent one instance of a CA, backed by a unique instance of the PKI Services daemon (and all its associated components), yet requiring no more than a single HTTP / WebSphere Application Server / Liberty Server.

Figure 34 on page 303 contains an illustration showing two CA domains, one for employees and one for customers. In the illustration, a single shared administrator supports both CA domains. (You can decide to share a common administrator across multiple CA domains or have separate administrators who are each dedicated to only one CA domain.)

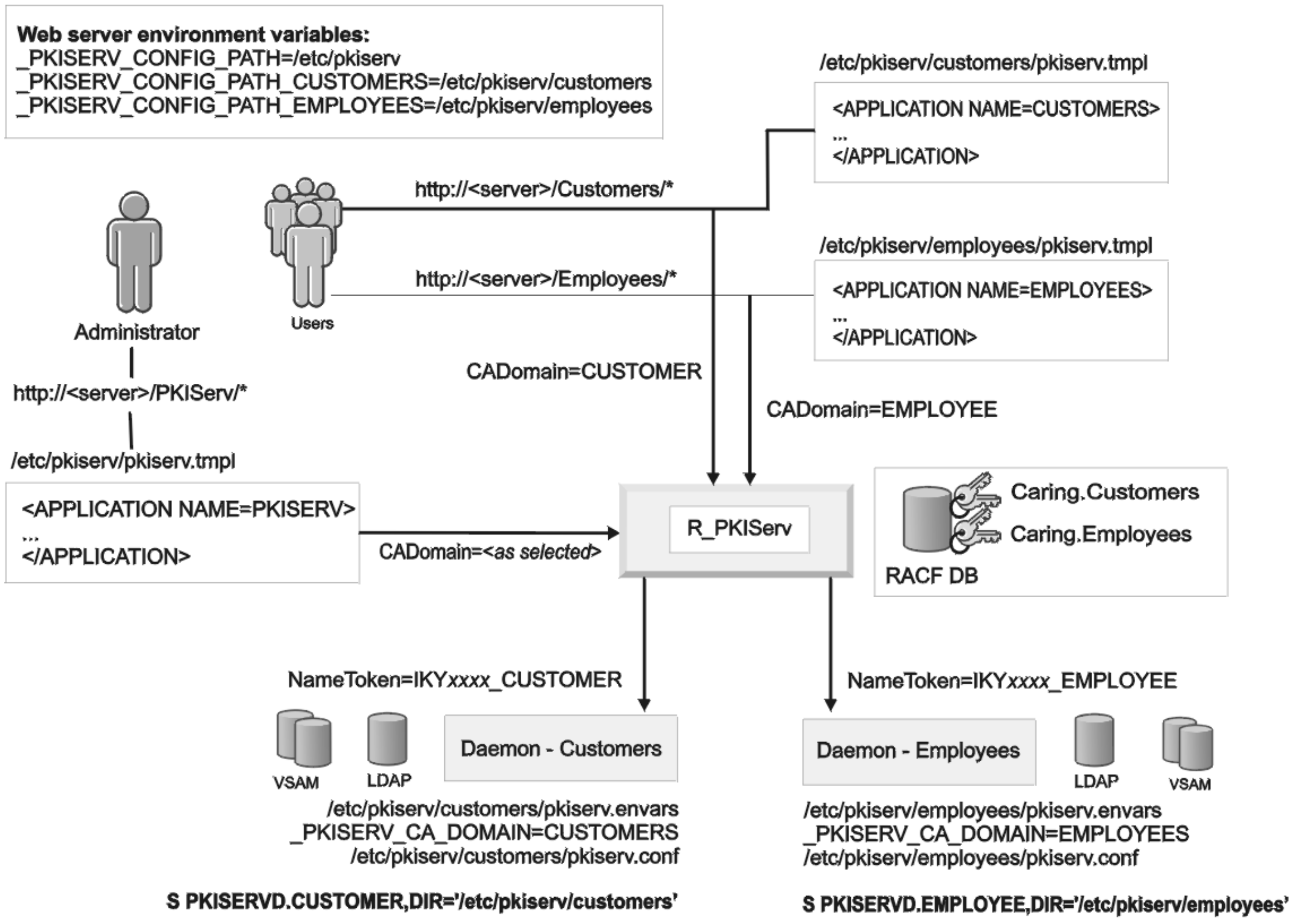

Web server environment variables:
_PKISERV_CONFIG_PATH=/etc/pkiserv
_PKISERV_CONFIG_PATH_CUSTOMERS=/etc/pkiserv/customers
_PKISERV_CONFIG_PATH_EMPLOYEES=/etc/pkiserv/employees
/etc/pkiserv/customers/pkiserv.tmpl
<APPLICATION NAME=CUSTOMERS>
...
</APPLICATION>
http://<server>/Customers/*
/etc/pkiserv/employees/pkiserv.tmpl
<APPLICATION NAME=EMPLOYEES>
...
</APPLICATION>
http://<server>/Employees/*
Administrator
Users
CADomain=CUSTOMER
CADomain=EMPLOYEE
http://<server>/PKIServ/*
/etc/pkiserv/pkiserv.tmpl
<APPLICATION NAME=PKISERV>
...
</APPLICATION>
CADomain=<as selected>
R_PKIServ
Caring.Customers
Caring.Employees
RACF DB
NameToken=IKYxxxx_CUSTOMER
NameToken=IKYxxxx_EMPLOYEE
VSAM
LDAP
Daemon - Customers
Daemon - Employees
LDAP
VSAM
/etc/pkiserv/customers/pkiserv.envars
_PKISERV_CA_DOMAIN=CUSTOMERS
/etc/pkiserv/customers/pkiserv.conf
/etc/pkiserv/employees/pkiserv.envars
_PKISERV_CA_DOMAIN=EMPLOYEES
/etc/pkiserv/employees/pkiserv.conf
S PKISERVD.CUSTOMER,DIR='/etc/pkiserv/customers'
S PKISERVD.EMPLOYEE,DIR='/etc/pkiserv/employees'

Figure 34. Illustration of two CA domains, one for employees and one for customers, administered by a single shared administrator who administers both domains

Task overview

This topic includes a task roadmap that you can use to add a new CA domain. The roadmap includes several subtasks, which are listed in Table 51 on page 304. It is intended to direct you to add a new CA domain after you have completed all required tasks in Part 2, "Configuring your system for PKI Services," on page 37. Before you begin this task, you have already implemented and tested the default setup for PKI Services and ensured that it operates properly as a single CA domain.

For each CA you add, you create a dedicated copy of the object store and issued certificate list (ICL), CA certificate, key ring, and LDAP namespace. You also create a dedicated copy of the PKI Services configuration file (`pkiserv.conf`), templates file (`pkiserv.tmpl` or `pkitmpl.xml`), and environment variables file (`pkiserv.envars`), each in its own directory. You update the following CA-specific information in these files:

- `pkiserv.conf` - contains the CA-specific key ring name, VSAM data set names or Db2 subsystem and package name for the object store and ICL, and optionally `CRLDistDirPath`.
- `pkiserv.envars` - contains a variable `_PKISERV_CA_DOMAIN` to specify CA domain and the variable `_PKISERV_CONFIG_PATH` sets the directory for each CA domain.

- The template file that you are using:
  - `pkiserv.tmpl` - contains the name of the end-user application section (default is CUSTOMER) that you rename to a CA-specific name, such as `<APPLICATION NAME=EMPLOYEE>`. It also contains the name of the administrative application section (default is PKISERV) that you can rename to a CA-specific name, such as `<APPLICATION NAME=ADMEMPLOYEES>`.
  - `pkitmpl.xml` - defines the applications and certificate request templates that you use for this CA domain.

If you are implementing the web application using REXX CGI execs, see "Subtask 2: Steps for reconfiguring your initial CA domain to allow it to coexist with other CA domains" on page 307 and "Subtask 6: Steps for updating the web server configuration" on page 314 for more information about adding application domains.

If you are implementing the web application using JavaServer pages (JSPs), you need to edit the `web.xml` file with the `PKIServ.EAR`. It helps to understand the URLs used for multiple CA domains and multiple application domains. The JSP code parses the URL to determine the CA domain and application name. The first directory after the root context of PKIServ_Web is the CA domain name if there is one and the second directory is either the application name or PKIServ (for the PKI Services administration web pages). When the JSPs are run without a named CA domain, the first directory after the root context of PKIServ_Web is the application name or PKIServ (for the PKI Services administration pages).

Task roadmap for adding CA domains

Before you begin

- Complete all required tasks in Part 2, "Configuring your system for PKI Services," on page 37. This task roadmap is intended to direct you to add a new CA domain *after* you have already implemented and tested the default setup for PKI Services and ensured that it operates properly as a single CA domain.
- Review Table 51 on page 304 to see the subtasks that are involved and the skills that are required for each subtask. (The team members that are listed are based on role definitions that are established in "Identifying skill requirements" on page 14.)

Procedure

To create a new CA domain, complete the subtasks in Table 51 on page 304. Subtask 1 guides you through planning. Subtask 2 is a one-time setup that you do when you add your first additional CA domain. Subtasks 3 - 8 are each done once for every CA domain you add. (See Part 2, "Configuring your system for PKI Services," on page 37 for additional details about these subtasks.)

After you complete Subtasks 1 and 2 (planning and reconfiguring), perform Subtasks 3 - 8 for your *first* new CA domain and ensure that it operates properly before adding your second CA domain.

Table 51. Task roadmap for adding a new CA domain

	Subtask	Team member	Associated procedure (See...)
1	Plan additional CA domains.	UNIX programmer	"Subtask 1: Steps for planning additional CA domains" on page 305.
2	Reconfigure your initial CA domain to allow it to coexist with other CA domains. (This is a one-time setup.)	UNIX programmer	"Subtask 2: Steps for reconfiguring your initial CA domain to allow it to coexist with other CA domains" on page 307.
3	Run IKYSETUP.	MVS programmer	"Subtask 3: Steps for running the IKYSETUP exec" on page 309.

Table 51. Task roadmap for adding a new CA domain (continued)

	Subtask	Team member	Associated procedure (See...)
4	Configure the UNIX environment.	UNIX programmer	“Subtask 4: Steps for configuring the UNIX environment” on page 311.
5	Update the PKI Services template file or JSP files.	Web server programmer	“Subtask 5: Steps for updating the PKI Services template file or JSP files” on page 312.
6	Update the web server configuration.	Web server programmer	“Subtask 6: Steps for updating the web server configuration” on page 314.
7	Set up the object store and ICL.	MVS programmer or Db2 programmer, depending on how you implement the object store and ICL	“Subtask 7: Creating the object store and ICL” on page 316.
8	Start PKI Services.	MVS programmer	“Subtask 8: Steps for starting PKI Services” on page 317.

Recording your progress adding CA domains

As you complete each subtask for adding a new CA domain, use Table 52 on page 305 to mark your progress. The tasks correspond to the subtasks listed in the roadmap in Table 51 on page 304.

Table 52. Multiple CA domains: Worksheet #1 for recording progress adding new CA domains

	Subtask	First CA domain	Second CA domain	Third CA domain
1	Plan additional CA domains.	☐		
2	Reconfigure initial domain. (once only)	☐		
3	Run IKYSETUP.	☐	☐	☐
4	Configure the UNIX environment.	☐	☐	☐
5	Update the PKI Services template file. (Perform this task only if you implement the web application using REXX CGI execs.)	☐	☐	☐
6	Update the web server configuration.	☐	☐	☐
7	Set up the object store and ICL.	☐	☐	☐
8	Start PKI Services.	☐	☐	☐

Guideline: Perform Subtasks 1 - 8 for your *first* new CA domain and ensure that it operates properly before adding your second CA domain.

Subtask 1: Steps for planning additional CA domains

Perform the following steps to plan additional CA domains.

1. Determine how many instances of PKI Services (CA domains) you operate in addition to the initial domain you configured when you originally customized PKI Services.

   For each CA domain, you need to pick a nickname to use as the CA domain name. The CA domain name is used to qualify the resources used by that CA domain. For example, the CA domain named `Employees` uses the following resources:

   Examples:

   - Web page URLs (in mixed case)
     - If you implement the web application using REXX CGI execs:

       ```
       https://webserver-domain-name/Employees/ssl-cgi/camain.rexx
       ```

     - If you implement the web application using JSPs:

       ```
       http://webserver-domain-name/PKIServ_Web/Employee/ApplicationName/pkimain.jsp
       ```

       Note that the CA domain name is independent of the application domain name.
   - Data set qualifiers (in uppercase) - VSAM ICL data set

     ```
     PKISRVD.EMPLOYEE.VSAM.ICL
     ```

   - Path names (in lowercase)

     ```
     /etc/pkiserv/employees/pkiserv.conf
     ```

   If you are implementing the object store and ICL using Db2, you need a unique Db2 package name for each CA domain. Use the CA domain name for the package name. You should also use a unique name for the Db2 plan.

   ____

2. Decide how you want to administer multiple CA domains. Will you share a *common* set of administrators across all your CA domains or will you have a *dedicated* set of administrators for each CA domain?

   If you use a *dedicated* set for each CA domain, you need to pick a second nickname for each CA domain, for its administrative domain.

   ____

3. Determine your CA domain names. Unless you renamed the default domain names when you originally customized PKI Services, the initial name for the application domain is `Customers` and its administrative domain name is `PKIServ`. Your new CA domain names (nicknames) must differ from these values.

   Rules for domain names:

   - Domain names are 1 - 8 characters.
     - For REXX CGI execs, domain names can exceed 8 characters if the first 8 characters are unique from your other domain names.
     - For JavaServer pages (JSPs) domain names cannot exceed 8 characters.
   - The characters in the domain name are limited to the following character set: alphanumeric characters (`a - z`, `A - Z`, `0 - 9`) and the hyphen (`-`).
   - The first character must not be a number or hyphen.

   ____

4. Record information about your CA domains in Table 53 on page 307 and Table 54 on page 307.

   Row **1** in each table is already filled in with the defaults for an initial CA domain (`Customers`). Row **2** in each table is an example of a new CA domain managed by the same (shared) group of administrators.

Row **3** in each table is an example of the same CA domain from Row **2** managed by a *dedicated* group of administrators.

The rows in each table that are already filled in use the default values for the following variables when PKI Services was installed. (Your MVS programmer might have chosen different directories.)

Installation variable	Default directory name
install-dir	`/usr/lpp/pkiserv`
runtime-dir	`/etc/pkiserv`

a. Fill in the values for new CA domains, administrative domains, and directories in Table 53 on page 307. You can add your information in the following blank lines or you can modify or cross out the sample rows.

Table 53. Multiple CA domains: Worksheet #2 for planning your domain names

	CA domain name (`runtime directory`)	Truncated CA domain name	Administrative domain name (`runtime directory`)
1	Customers (`/etc/pkiserv`)	CUSTOMER	PKIServ (`/etc/pkiserv`)
2	Employees (`/etc/pkiserv/employees`)	EMPLOYEE	PKIServ (`/etc/pkiserv`)
3	Employees (`/etc/pkiserv/employees`)	EMPLOYEE	AdmEmployees (`/etc/pkiserv/employees`)
4			
5			

b. Fill in your RACF user IDs, groups, and VSAM data set qualifiers or Db2 package names in Table 54 on page 307. You can add your information in the following blank lines or you can modify or cross out the sample rows.

Table 54. Multiple CA domains: Worksheet #3 for planning your RACF identifiers, z/OS UNIX identifiers, and VSAM data set names or Db2 package names. Use row 2 for shared administrators, row 3 for dedicated administrators. Row 4 is the same as row 3 but specifies a Db2 package name instead of VSAM data set qualifiers.

	Daemon user ID (UID)	Surrogate user ID (UID)	PKI administration group name (GID)	VSAM data set qualifiers or Db2 package name
1	PKISRVD (554)	PKISERV (555)	PKIGRP (655)	PKISRVD.VSAM
2	PKISRVD (554)	PKISERV (555)	PKIGRP (655)	PKISRVD.EMPLOYEE.VSAM
3	PKIDEMP (556)	PKISEMP (557)	PKIGEMP (657)	PKISRVD.EMPLOYEE.VSAM
4	PKIDEMP (556)	PKISEMP (557)	PKIGEMP (657)	MasterCA
5				

Subtask 2: Steps for reconfiguring your initial CA domain to allow it to coexist with other CA domains

Perform the following steps to reconfigure your initial CA domain to allow it to coexist with other CA domains. (This is a one-time setup that suffices no matter how many CA domains you add.)

1. If PKI Services is running, stop it by issuing the following MVS console command:

```
P PKISERVD
```

____

2. Update the PKI Services environment variables in the `pkiserv.envars` file as follows.
   a. (Optional) If your initial CA domain does not use its own `pkiserv.envars` file, copy the default `pkiserv.envars` file from the PKI Services install directory by issuing the following command from the UNIX command line:

      ```
      cp -p /usr/lpp/pkiserv/samples/pkiserv.envars /etc/pkiserv
      ```

   b. Edit the new copy of `pkiserv.envars` file by entering the following command:

      ```
      oedit /etc/pkiserv/pkiserv.envars
      ```

   c. Add a PKI Services environment variable identifying your initial CA domain name (see Table 53 on page 307) in uppercase characters.

      Example:

      ```
      _PKISERV_CA_DOMAIN=CUSTOMERS
      ```

____

3. Update the HTTP server's environment variables and configuration directives as follows.
   a. Update the HTTP server's environment variables. This setting is shown by adding it to the `vhost80.conf` (host file for non-SSL requests) by using the SetEnv HTTP directive. Similar changes must be made to the `vhost443` (host file for SSL requests with server authentication) and `vhost1443` (host file for SSL requests with client authentication) configuration files.
      1) Edit the `vhost80.conf` file by entering the following command:

         ```
         oedit /etc/websrv1/conf/vhost80.conf
         ```

      2) Add an environment variable identifying the runtime directory of your initial CA domain. (Check Table 53 on page 307.)

         Example:

         ```
         SetEnv _PKISERV_CONFIG_PATH_CUSTOMERS "/etc/pkiserv"
         ```

      3) (Optional) If you intend to have a dedicated set of administrators for each CA domain, add an environment variable that specifies the runtime directory for the administrative domain. (Check Table 53 on page 307.)

         Example:

         ```
         SetEnv _PKISERV_CONFIG_PATH_PKISERV "/etc/pkiserv"
         ```

____

4. Update the RACF access controls for the `R_PKIServ` SAF callable service as follows. (Any change to environment variables in Step "3" on page 308 requires a corresponding change to RACF access control.)
   a. Determine the PKI Services surrogate user ID (default is PKISERV) and the PKI Services administrators group (default is PKIGRP). To do this, refer to the log file created when the IKYSETUP REXX exec was originally run for your initial CA domain.
   b. Execute the following RACF commands from the TSO command line. Replace the highlighted values with your own, if different:

      Examples:

      ```
      RDELETE FACILITY IRR.RPKISERV.**
      RDEFINE FACILITY IRR.RPKISERV.*.CUSTOMER
      PERMIT IRR.RPKISERV.*.CUSTOMER CLASS(FACILITY) ID(PKISERV) ACCESS(CONTROL)
      RDELETE FACILITY IRR.RPKISERV.PKIADMIN
      RDEFINE FACILITY IRR.RPKISERV.PKIADMIN.CUSTOMER
      PERMIT IRR.RPKISERV.PKIADMIN.CUSTOMER CLASS(FACILITY) ID(PKIGRP)
           ACCESS(UPDATE)
      ```

```
PERMIT IRR.RPKISERV.PKIADMIN.CUSTOMER CLASS(FACILITY) ID(PKISERV)
     ACCESS(NONE)
SETROPTS RACLIST(FACILITY) REFRESH
```

Restriction: If the name of your initial CA domain is longer than 8 characters, you must truncate it to exactly 8 characters when you define the resource name in the FACILITY class profiles. (In this example, the name CUSTOMERS was truncated to CUSTOMER in the second RDEFINE FACILITY command.)

____

5. (Optional) You have reconfigured your initial CA domain to allow it to coexist with other CA domains. If you want, you can test the reconfiguration now. To test it, follow these steps:

   a. Restart PKI Services using the following MVS console command. Replace the highlighted values with your own, if different.

      Guideline: To simplify your environment, start this instance of PKI Services using a JOBNAME that matches the truncated name of the CA domain. (See your truncated value in Table 53 on page 307.) If you use the truncated values as job names, it is easier to distinguish multiple jobs that run PKI Services after you add other CA domains.

      Example:

      ```
      S PKISERVD,JOBNAME=CUSTOMER,DIR='/etc/pkiserv/'
      ```

   b. Restart the HTTP servers to enable your environment variable changes.

      ```
      F IMWEBSRV,APPL=-restart
      ```

   c. Test that your PKI Services application is functioning properly. Go to your web pages by entering the following URL from your browser:

      ```
      https://webserver-fully-qualified-domain-name/PKIServ/ssl-cgi/camain.rexx
      ```

      The *webserver-fully-qualified-domain-name* is the common name (CN) portion of the web server's distinguished name; see Table 11 on page 41.

      You should be able to go through your web pages to request, retrieve, and revoke a certificate of type "PKI browser certificate for authenticating to z/OS". Ensure you can do this before adding an additional CA domain.

____

When you are done: You have successfully reconfigured your initial CA domain to allow it to coexist with other CA domains. You can now perform each of the remaining subtasks once for each new CA domain.

Continue to the next subtask. **Guideline:** Complete Subtasks 3 - 8 for your *first* new CA domain and ensure that it operates properly before adding another CA domain.

Subtask 3: Steps for running the IKYSETUP exec

Before you begin

This procedure requires you to be familiar with the information in Chapter 4, “Running IKYSETUP to perform RACF administration,” on page 39. There are more details about the following steps there.

Procedure

Perform the following steps to customize a unique execution of the IKYSETUP REXX exec for this new CA domain.

1. Locate the IKYSETUP exec that you originally customized for your initial CA domain and copy it to a data set member that you can edit.

____

2. Edit the new copy of IKYSETUP and set the `ca_domain` variable to the name of this new CA domain. Type the domain name preserving the case of each character as you want it to appear in web page URLs.

   ____

3. If you intend to have a dedicated set of administrators for each CA domain, customize the following variables with your values for this CA domain.

Variable name	**Use your value from ...**
`daemon_uid`	Table 11 on page 41
`pki_gid`	Table 11 on page 41
`pkigroup_mem`	Table 11 on page 41
`surrog_uid`	Table 11 on page 41
`daemon`	Table 19 on page 55
`surrog`	Table 19 on page 55
`pkigroup`	Table 53 on page 307 (Use the truncated name of the administrative domain.)

   ____

4. (Optional) If you are creating multiple CAs as part of a certificate hierarchy where a previous CA domain is to be superior (as issuer or signer) of this CA domain, set `signing_ca_label` to match the label of the certificate in RACF that issues the certificate for this CA domain.

   Otherwise, skip to Step "5" on page 310 and leave `signing_ca_label=""` (the default).

   ____

5. Update any other values, such as `ca_dn` and `ra_dn`, that you choose to differ from your initial settings or the defaults.

   You need *not* change any values in this step unless you choose to set these values to something particular for your installation. (This is because when you specify a `ca_domain` value, the IKYSETUP exec automatically qualifies any value that PKI Services requires to be unique by adding the CA domain name.)

   ____

6. Execute IKYSETUP by entering the following TSO command:

```
EX 'data-set-name(new-member-name)' 'RUN(NO)'
```

   ____

7. Review the log data set to ensure that the commands created by IKYSETUP match your expectations. (For more information about these commands, see "Actions IKYSETUP performs by issuing RACF commands" on page 619.) Edit again as needed and rerun.

   ____

8. When you are satisfied with the commands and information in the log data set, rerun the IKYSETUP exec by entering the following TSO command:

```
EX 'data-set-name(new-member-name)' 'RUN(YES)'
```

   ____

9. Check your IKYSETUP log and record the name of the SAF key ring (your `ca_ring` value).

```

Name of the SAF key ring:
```

When you are done: You have customized and run the IKYSETUP exec for this CA domain. Record your progress in Table 52 on page 305.

Continue to the next subtask. **Guideline:** Complete all subtasks for this new CA domain and ensure that it operates properly before adding another CA domain.

Subtask 4: Steps for configuring the UNIX environment

Before you begin

This procedure requires you to be familiar with the information in Chapter 5, “Configuring the UNIX runtime environment,” on page 67. There are more details about the following steps there.

Procedure

Perform the following steps to configure the UNIX environment for this new CA domain.

1. Set up a `var` directory for this CA domain. Perform the steps in “ Steps for setting up the var directory” on page 95.
2. Locate the `pkiserv.conf`, `pkiserv.envars`, and `pkiserv.tmpl` files you originally used to create your initial CA domain. Copy them into the appropriate runtime directory for your new CA domain. (Check Table 53 on page 307.) For a new CA domain called `Employees`, run the following commands from the UNIX command line. (You might have to make the directory first.)

   Examples:

   ```
   mkdir /etc/pkiserv/employees
   chown pkisrvd /etc/pkiserv/employees
   cp -p /etc/pkiserv/* /etc/pkiserv/employees
   ```

   ---
3. Edit the new `pkiserv.conf` file by entering the following command:

   Example:

   ```
   oedit /etc/pkiserv/employees/pkiserv.conf
   ```

   ---
4. Change the following sections of `pkiserv.conf` as described for this CA domain. (Find detailed information for each variable in Table 21 on page 74.)

   ObjectStore

   If you are implementing the object store and ICL using VSAM, qualify each VSAM data set name with the CA domain name. **Example:** `ObjectDSN='pkisrvd.employee.vsam.ost'`

   If you are implementing the object store and ICL using Db2, set the Db2 package name to the CA domain name. **Example:** `DBPackage=employee`

   (See “Subtask 7: Creating the object store and ICL” on page 316.)

   CertPolicy

   If `CRLDistDirPath` is not null, modify it to reference the correct subdirectory. (You might have to create this directory.) **Example:** `CRLDistDirPath=/var/pkiserv/employees`. See “Determining CRLDistDirPath” on page 293 for more information.

   General

   Update each path name to the correct subdirectory. **Example:** `ReadyMessageForm=/etc/pkiserv/employees/readymsg.form`

   SAF

   Update the key ring name to match the `ca_ring` value you recorded. **Example:** `PKISRVD/Caring.Employees`

LDAP

Do not update the **LDAP** section unless you need to change the LDAP directory. If you need to change it, see "Steps for tailoring the LDAP section of the configuration file" on page 108.

Make sure that the LDAP directory is configured with a suffix for this CA domain. (See the explanation for the `Suffix` variable in Table 22 on page 97.)

____

5. (Optional) Change other values in any section of `pkiserv.conf` as you want for this CA domain.

____

6. Edit the new `pkiserv.envars` file by entering the following command:

   Example:

   ```
   oedit /etc/pkiserv/employees/pkiserv.envars
   ```

____

7. Define the `_PKISERV_CA_DOMAIN` environment variable for this CA domain name. (For details, see "The pkiserv.envars environment variables file" on page 617.)

   Example:

   ```
   _PKISERV_CA_DOMAIN=EMPLOYEE
   ```

____

When you are done: You have updated the `pkiserv.conf` and `pkiserv.envars` files for this CA domain. Record your progress in Table 52 on page 305.

Continue to the next subtask. **Guideline:** Complete all subtasks for this new CA domain and ensure that it operates properly before adding another CA domain.

Subtask 5: Steps for updating the PKI Services template file or JSP files

Update the PKI Services template file if you implement the web application using REXX CGI execs. Update the JSP files if you implement the web application using JSPs.

Before you begin

This procedure requires you to be familiar with the information in Chapter 11, "Customizing the end-user web application if you use REXX CGI execs," on page 135 if you use the HTTP Server; or information in Chapter 13, "Implementing the web application using JavaServer pages," on page 237 if you use the Traditional WebSphere Application Server or Liberty Server.

Procedure to customize the PKI Services template file

Perform the following steps to customize the PKI Services template file (`pkiserv.tmpl`) for this new CA domain.

1. Edit the new `pkiserv.tmpl` file (you copied it in Step "2" on page 311 of "Subtask 4: Steps for configuring the UNIX environment" on page 311) by entering the following command from the UNIX command line:

   Example:

   ```
   oedit /etc/pkiserv/employees/pkiserv.tmpl
   ```

____

2. Locate all occurrences of the CA domain named `Customers` (in mixed case, uppercase, or lowercase) and change them to the name of this new CA domain, being careful to preserve the case.

   Example: If the new CA domain is `Employees` (in mixed case), change the default values to the name of your new CA domain.

Default values for the `Customers` CA domain	**New values for the `Employees` CA domain**
`ACTION="/Customers/…"` (in mixed case)	`ACTION="/Employees/…"` (also in mixed case)
`<APPLICATION NAME=CUSTOMERS>` (in uppercase)	`<APPLICATION NAME=EMPLOYEES>` (also in uppercase)

____

3. If you intend to have the *same* set of administrators for all your CA domains, *skip* this step and proceed to Step "4" on page 313.

   If you intend to have a *dedicated* set of administrators for each CA domain, change the name of the PKISERV application section to the corresponding name of the administrative domain in Table 53 on page 307. This value must be specified in uppercase characters only.

   Example: If the new administrative domain is named `AdmEmployees`, change the default value (PKISERV) to the name of your new administrative CA domain.

Default value for the PKISERV administrative CA domain	**New value for the new `AdmEmployees` administrative CA domain**
`<APPLICATION NAME=PKISERV>` (in uppercase)	`<APPLICATION NAME=ADMEMPLOYEES>` (also in uppercase)

____

4. If you intend to have the same set of administrators for all your CA domains, edit the main templates file (`/etc/pkiserv/pkiserv.tmpl`) as follows:

   Do not update the PKISERV application section in this domain-specific `pkiserv.tmpl` file; it is *not* used. The PKISERV application section in the templates file for your initial CA domain is used instead.

   a. Replicate the following lines in the APPLICATION section of the PKISERV application:

```
<h3>Go the Customers' home page </h3>
<FORM name=admform METHOD=GET
   ACTION="/Customers/ssl-cgi/camain.rexx">
<p>
<INPUT TYPE="submit" VALUE="Customers' Home Page">
</FORM>
```

   b. Change all occurrences of the string `Customers` in the replicated lines to the name of this CA domain, being careful to preserve case.

      Example: Change `ACTION="/Customers/ssl-cgi/camain.rexx">` to `ACTION="/Employees/ssl-cgi/camain.rexx">`.

   c. Uncomment the `%%SelectCADomain%%` directive in the ADMINSCOPE subsection of the APPLICATION section for the PKISERV application by removing the leading # character. (The `%%SelectCADomain%%` directive enables multiple CA administration.)

   d. Update the `SelectCADomain` insert to include an OPTION entry for this CA domain. If this CA domain is used more often than any other, mark the entry SELECTED and remove SELECTED from any other entry.

      Example:

```
<INSERT NAME=SelectCADomain>
<p> <LABEL for="selectcadomfield">Select the CA domain to work with </LABEL>
```

```
<SELECT NAME="domain" id="selectcadomfield">
# rename and replicate the following line for every CA domain and
# determine which one should be SELECTED by default, if any
<OPTION VALUE="Employees" SELECTED>Employees
<OPTION VALUE="Customers">Customers
</SELECT>
```

Procedure to customize the JavaServer page files

You must have a new PKI Services EAR file containing the customized JSP files for the new CA domain. You can follow the steps for Traditional WebSphere Application Server or for Liberty in Chapter 13, "Implementing the web application using JavaServer pages," on page 237.

When you are done:
You have customized the PKI Services template file (`pkiserv.tmpl`) or JavaServer page files for this CA domain. Record your progress in Table 52 on page 305.

Continue to the next subtask. **Guideline:** Complete all subtasks for this new CA domain and ensure that it operates properly before adding another CA domain.

Subtask 6: Steps for updating the web server configuration

If you are implementing the web application using REXX CGI execs, perform the steps in "Updating the web server configuration if you use REXX CGI execs" on page 314

If you are implementing the web application using JavaServer pages (JSPs), perform the steps in "Updating the web server configuration if you use JavaServer pages (JSPs)" on page 315

Updating the web server configuration if you use REXX CGI execs

Before you begin

This procedure requires you to be familiar with the information in the following topics, where you find additional details:

- "Steps for adding application domains to the web server configuration files" on page 300
- Chapter 7, "Updating IBM HTTP Server - Powered by Apache configuration and starting the server," on page 101.

Procedure

Perform the following steps to customize the IBM HTTP Server configuration files for this new CA domain.

1. Add this new CA domain (check Table 53 on page 307 for domain name and directory) following the instructions in "Steps for adding application domains to the web server configuration files" on page 300. If you have a `CRLDistDirPath` configured in your `pkiserv.conf` file for HTTP protocol URI format CRL distribution points, uncomment the AliasMatch HTTP server directive statement in the `vhost80.conf` (host file for non-SSL requests) configuration file to map the virtual path name in the URI to the `CRLDistDirPath` value.

   ```
   #AliasMatch /Employees/cacerts/(.*) /var/pkiserv/employees/$1
   ```

2. (Optional) If you intend to have a dedicated set of administrators for each CA domain, repeat Step "1" on page 314 for the administrative domain. (Check Table 53 on page 307 for domain name and directory.) Otherwise, skip to Step "3" on page 314.
3. Update the environment variables for the HTTP Server. The following changes must be made to the `vhost80.conf` (host file for non-SSL requests), `vhost443.conf` (host file for SSL requests with server authentication), and `vhost1443.conf` (host file for SSL requests with client authentication)

configuration files. The setting is shown by using the `SetEnv` HTTP directive in the `vhost80.conf` file. Edit the `vhost80.conf` file by entering the following command from the UNIX command line:

```
oedit /etc/websrv1/conf/vhost80.conf
```

Uncomment the `SetEnv` HTTP directive statement to add the environment variable identifying the runtime directory of this CA domain. (Check Table 53 on page 307.)

```
#SetEnv _PKISERV_CONFIG_PATH_EMPLOYEES "/etc/pkiserv/employees"
```

4. (Optional) If you intend to have a dedicated set of administrators for each CA domain, add the environment variable identifying the `pkiserv.tmpl` directory of this administrative CA domain.

   Uncomment `SetEnv` HTTP directive statement.

   ```
   #SetEnv _PKISERV_CONFIG_PATH_ADMEMPLOYEES "/etc/pkiserv/employees"
   ```

5. Update the HTTP server virtual host configuration files to support the new CA Domain. Uncomment the following statements in each of the virtual host configuration files.

   `vhost80.conf`

   ```
   • #RewriteRule ^/(AdmEmployees|Employees)/ssl-cgi/(.*) https://<server-domain-name>/$1/ssl-cgi-bin/$2 [R,NE]
   • #RewriteRule ^/(AdmEmployees|Employees)/clientauth-cgi/(.*) https://<server-domain-name>:1443/$1/clientauth-cgi-bin/$2
   [R,NE]
   • #ScriptAliasMatch /(AdmEmployees|Employees)/public-cgi/(.*) <application-root>/PKIServ/public-cgi/$2
   • #AliasMatch /Employees/cacerts/(.*) /var/pkiserv/$1
   ```

   `vhost443.conf`

   ```
   • #RewriteRule ^/(AdmEmployees|Employees)/public-cgi/(.*) http://<server-domain-name>/$1/public-cgi/$2 [R,NE,L]
   • #RewriteRule ^/(AdmEmployees|Employees)/ssl-cgi/(.*) https://<server-domain-name>/$1/ssl-cgi-bin/$2 [R,NE]
   • #RewriteRule ^/(AdmEmployees|Employees)/clientauth-cgi/(.*) https://<server-domain-name>:1443/$1/clientauth-cgi-bin/$2
   [R,NE,L]
   • #ScriptAliasMatch ^/(AdmEmployees|Employees)/(public-cgi|ssl-cgi-bin)/(.*) "<application-root>/PKIServ/$2/$3"
   • #<LocationMatch "^/(AdmEmployees|Employees)/ssl-cgi-bin(/(auth|surrogateauth))?/cagetcert.rexx">
       # Charsetoptions TranslateAllMimeTypes
     #</LocationMatch>
   ```

   `vhost1443.conf`

   ```
   • #RewriteRule ^/(AdmEmployees|Employees)/public-cgi/(.*) http://<server-domain-name>/$1/public-cgi/$2 [R,NE,L]
   • #RewriteRule ^/(AdmEmployees|Employees)/ssl-cgi/(.*) https://<server-domain-name>/$1/ssl-cgi-bin/$2 [R,NE,L]
   • #ScriptAliasMatch ^/(AdmEmployees|Employees)/(clientauth-cgi|clientauth-cgi-bin)/(.*)
       "<application-root>/PKIServ/clientauth-cgi-bin/$3"
   • #<LocationMatch "^/(AdmEmployees|Employees)/clientauth-cgi-bin/auth/pkicmp">
       #CharsetOptions NoTranslateRequestBodies
     #</LocationMatch>
   ```

When you are done: You have customized the IBM HTTP Server configuration files for this CA domain. Record your progress in Table 52 on page 305.

Continue to the next subtask. Complete all subtasks for this new CA domain and ensure that it operates properly before adding another CA domain.

Updating the web server configuration if you use JavaServer pages (JSPs)

To add a new CA domain when you implement the web application using JavaServer pages (JSPs), you need to edit the `web.xml` file to define additional environment variables and URL mappings to the WebSphere application, `PKIServ_EAR`. The shipped `web.xml` file contains the definitions to use either an unnamed domain or a domain named "Master". Use the definition of the CA domain Master as a model for defining additional CA domains. This section in the `web.xml` file is delimited by the two comments:

```
<!-- Start: For new domain: Master -->
⋮
<!-- End: For named domain Master with application Customers: Master/Customers -->
```

If you use the Traditional WebSphere Application Server, you must deploy the new EAR file that you created in “Subtask 5: Steps for updating the PKI Services template file or JSP files” on page 312 using the WebSphere administrative console. If you use the Liberty Server, you must create a new `enterpriseApplication` pointing to the new EAR file in the same `server.xml` file, or configure it in a separate `server.xml` file if you want to run the new CA domain in a separate server.

Subtask 7: Creating the object store and ICL

Before you begin

This procedure requires you to be familiar with the information in Chapter 9, "Creating the object store and ICL," on page 115. There are more details about the following steps there.

You can implement the object store and ICL using either VSAM data sets or Db2 tables.

Procedure

If you are implementing the object store and ICL using VSAM data sets, perform the following steps to allocate the needed VSAM files for this new CA domain.

1. Locate the IKYCVSV1 JCL job that you originally customized for your initial CA domain and copy it to a data set that you can edit.

   ____

2. Change all occurrences of `PKISRVD.VSAM` to include the corresponding VSAM data set qualifier from Table 54 on page 307. For example, if this new CA domain is `Employees`, then the VSAM data sets might be qualified as follows:

   Example:

   ```
   PKISRVD.EMPLOYEE.VSAM.data-set-suffix
   ```

   ____

3. Submit the JCL job when your changes are complete.

   ____

When you are done: You have allocated the needed VSAM files for this CA domain. Record your progress in Table 52 on page 305.

If you are implementing the object store and ICL using Db2 tables, perform the following steps to create the needed Db2 tables, package, and plan.

1. Locate the IKYCDBV1 sample that you originally customized for your initial CA domain and copy it to a data set that you can edit.

   ____

2. Change all occurrences of `MASTERCA` to the package name from Table 54 on page 307. For example, if this new CA domain is `Employees`, and you are using the domain name for the package name, the table created for the object store would be named `EMPLOYEE.OST`.

   ____

3. When your changes are complete, run the modified sample using SPUFI.

   ____

4. Locate the IKYSBIND sample job that you originally customized for your initial CA domain and copy it to a data set that you can edit.

   ____

5. Change all occurrences of `MASTERCA` to the package name from Table 54 on page 307.

   ____

6. Submit the JCL job when your changes are complete.

   ____

When you are done: You have create the needed Db2 tables, package, and plan for this CA domain. Record your progress in Table 52 on page 305.

Continue to the next subtask. **Guideline:** Complete all subtasks for this new CA domain and ensure that it operates properly before adding another CA domain.

Subtask 8: Steps for starting PKI Services

Before you begin

This procedure requires you to be familiar with the information in Chapter 10, "Starting and stopping PKI Services," on page 129. There are more details about the following steps there.

Procedure

Perform the following steps to start a separate instance of PKI Services for this new CA domain.

1. Start the PKI Services daemon for this CA domain by entering the MVS console START command qualified with the appropriate runtime directory. (Check Table 53 on page 307.)

   Example:

   ```
   S PKISERVD,JOBNAME=EMPLOYEE,DIR='/etc/pkiserv/employees'
   ```

   Guideline: To simplify your environment, give this instance of PKI Services a JOBNAME that matches or relates to this CA domain name. When you add additional CA domains, it is easier to distinguish multiple jobs running PKI Services.

   ____

2. Restart the HTTP servers to enable the environment variables you changed for this CA domain. Optionally, you can wait to do this until after you have started all the new domain-specific daemons.

   ```
   S WEBSRV1,ACTION='stop'
   ```

   Then

   ```
   S WEBSRV1
   ```

   ____

3. Test that your new domain-specific PKI Services daemon is functioning properly. Go to your web pages by entering the following URL from your browser:

   ```
   https://<webserver-fully-qualified-domain-name>/<new-admin-domain-name>/ssl-cgi/camain.rexx
   ```

   The *webserver-fully-qualified-domain-name* is the common name (CN) portion of the web server's distinguished name; see Table 11 on page 41.

   You should be able to go through your web pages to request, retrieve, and revoke an applicable certificate for this CA domain, possibly "PKI browser certificate for authenticating to z/OS". Ensure you can do this before adding new CA domains.

   ____

When you are done: You have customized the IBM HTTP Server configuration files for this CA domain. Record your progress in Table 52 on page 305.

Once your new CA domain works properly, proceed to add another CA domain, if needed. **Guideline:** Perform Subtasks 3 - 8 for each new CA domain and ensure that the new CA domain operates properly before proceeding to add another.

Customizing email notifications sent to users

You can optionally notify a user by sending an email message when:

- A certificate request is rejected

- A certificate is ready for retrieval
- A certificate is about to expire (unless it is already renewed or revoked).
- A certificate is automatically renewed.
- Requests are pending for the user's approval, when the user is the PKI administrator.

In addition, when a user requests that PKI Services recover one or more certificates for which PKI Services created the keys, PKI Services sends that user an email message listing the possible certificates to be recovered.

On the days that are specified by the `MaintRunDays` parameter in the `pkiserv.conf` file, the PKI Services daily maintenance task checks the issued certificate list (ICL) for expiring certificates. (The ExpireWarningTime parameter (see the **CertPolicy** section in Table 21 on page 74) determines at what point before the certificate expires that it is considered to be an expiring certificate.) When PKI Services finds an expiring certificate, it takes one of the following actions:

- If automatic renewal of certificates is in effect, PKI Services renews the certificate and sends it to the client.
- If automatic renewal of certificates is not in effect, PKI Services sends an expiration warning message to the client (unless the certificate is already revoked). Regardless of whether sending the expiration warning message is successful, PKI Services makes only one attempt to send a notification message. If the email address is incorrect or the user renews the certificate and retrieves it before the expiration message is sent, no expiration messages are sent.

You can set the AdminNotifyNew*n* keyword in the CertPolicy section of the configuration file to specify one or more email addresses of PKI administrators to be notified immediately whenever there is a request pending for approval. The notification is sent only once, when the request is created. You can also set the AdminNotifyReminder*n* keyword to specify one or more email addresses of PKI administrators to be reminded once a day of any requests pending for approval. An administrator receives a daily reminder of a pending request until the request is processed. To receive both the immediate notifications and the daily reminders, an administrator's email must be specified on both the AdminNotifyNew and AdminNotifyReminder keywords.

If you are not sending email notifications, see Step “6.b” on page 220 for directions.

If you are sending email notifications, you need to do the following things:

- Have copies of the forms in the runtime directory. (For information about copying the message forms to the runtime directory, see Step “3” on page 70.
- Customize the forms. (For details, see “Steps for customizing email notification forms” on page 322.)
- For notifications that a request is rejected, that a certificate is ready for retrieval, that a certificate is about to expire, and that a certificate is renewed automatically, include the `NotifyEmail` field on certificate requests. This field is already included in the `pkiserv.tmpl` certificate template file. If you are *not* sending email notifications, you need to delete the `NotifyEmail` lines in the `pkiserv.tmpl` file; for details, see Step “6.b” on page 220.)

  For more information about the NotifyEmail field, see Table 32 on page 139. For information about fields on request forms, see Table 66 on page 379.

The following examples (of notices you can send to users) are in the sample directory:

```
From:dime-o-cert PKI
Subject:Certificate Ready For Pick Up
Attention - Please do not reply to this message as it was automatically sent
by a service machine.
Dear %%requestor%%,
Thank you for choosing dime-o-cert PKI. The certificate you requested
for subject %%dn%% is now ready for pickup.
Please visit:
(edit the following link if using the REXX CGI Web Application)
https://www.dimeocert.com/Customers/ssl-cgi-bin/camain.ress
(edit the following link if using the JSP WebSphere Application)
https://www.dimeocert.com:9943/PKIServ_Web/Customers/certretrieve.jsp?%%quicklink%%
to retrieve your certificate.
If that link does not work, try to go to
(edit the following link if using the REXX CGI Web Application)
https://www.dimeocert.com/Customers/ssl-cgi/camain.rexx
(edit the following link if using the JSP WebSphere Application)
https://www.dimeocert.com:9080/PKIServ_Web/Customers/pkimain.jsp
And enter the transaction ID listed below:
%%transactionid%%  You will need to input your passphrase that you
entered when you submitted the request.
```

Figure 35. Sample of readymsg.form

```
From:dime-o-cert PKI
Subject:Certificate Request Rejected

Attention - Please do not reply to this message as it was
automatically sent by a service machine.

Dear %%requestor%%,

Thank you for choosing dime-o-cert PKI. We are sorry to inform you that
your certificate request for subject %%dn%% has been rejected
with the following explanation [if any]:
%%rejectreason%%.
Please contact the PKI Services administrator at 1-800-xxx-xxxx.
You will need the transaction ID listed below.

%%transactionid%%
```

Figure 36. Sample of rejectmsg.form

```
From:dime-o-cert PKI
Subject:Certificate Expiration
Attention - Please do not reply to this message as it was automatically sent by a service machine.
Dear %%requestor%%,
Thank you for choosing dime-o-cert PKI. The certificate your requested for
subject %%dn%% expires at %%notafter%% local time. If you wish to renew
your certificate, please visit:
(edit the following link if using the REXX CGI Web Application)
https://www.dimeocert.com/Customers/ssl-cgi-bin/camain.rexx
(edit the following link if using the JSP WebSphere Application)
https://www.dimeocert.com:9443/PKIServ_Web/Customers/pkimain.jsp
If this is a browser certificate, you must use the same workstation and browser that
you used when you requested the original certificate. If this is a server
certificate, you will have to submit a #10 certificate request.
```

Figure 37. Sample of expiringmsg.form

```
From:dime-o-cert PKI
Subject:Certificate Renewed
Attention - Please do not reply to this message as it was automatically
sent by a service machine.
Dear %%requestor%%,
Your certificate with subject name %%dn%% has been automatically renewed.
Here is your new certificate in Base64 encoded format: %%printcert%%
If your original certificate is installed in Internet Explorer,
click on this link to install the above new certificate:
(edit the following link if using the REXX CGI Web Application)
https://www.dimeocert.com/Customers/ssl-cgi-bin/installcert.rexx
(edit the following link if using the JSP WebSphere Application)
https://www.dimeocert.com:9443/PKIServ_Web/Customers/installcert.jsp
```

Figure 38. Sample of renewcertmsg.form

Note: Port number 9443 should match your secure port address (`_PKISERV_SECURED_PORT`) in the `web.xml` file.

```
From:dime-o-cert PKI
Subject:Request(s) pending for approval

Dear %%cadomain%% administrator,
The following request(s) is/are waiting for your approval:
%%pendreqlist%%
```

Figure 39. Sample of pendingmsg.form

```
From:dime-o-cert PKI
Subject:Certificate Requests are modified, reapproval is needed

Dear %%cadomain%% administrator,

The following requests, waiting for your approval, are modified
by the specified administrators. Previous approvals are invalidated.

Request Requestor Approvals Modified by
                  Required
%%modreqlist%%
```

Figure 40. Sample of pendingmsg2.form

```
From:dime-o-cert PKI
Subject:Certificate Recovery
Attention - Please do not reply to this message as it was automatically sent
by  a service machine.
Dear %%requestor%%,
Here is a list of certificate(s) that satisfy your searching criteria for
recovery: %%recoverylist%%  Please choose the certificate you want and
visit the corresponding link to  retrieve it
(you can identify the certificate by the serial number from the  part of the
link between '?' and '&')   (edit the following link if using the REXX CGI Web Application)
https://www.dimeocert.com/Customers/ssl-cgi-bin/caretrieve.rexx?%%recoverylink%%
(edit the following link if using the JSP WebSphere Application)
https://www.dimeocert.com:9080/PKIServ_Web/Customers/certrecover.jsp?%%recoverylink%%
You will need to input your pass phrase that you entered when you submitted the  request.
```

Figure 41. Sample of recoverymsg.form

Notes:

1. PKI Services automatically provides the `To:` value in the forms. You can include `From:` or `Subject:` or both at the beginning of the file.
2. You must have a blank line between the Subject and the body of the form.

The following table summarizes the variables that you can use in the forms when you customize them. At run time, PKI Services replaces these with their actual values.

Table 55. Descriptions of variables for forms

Variable	Description
%%cadomain%%	The CA domain that this message comes from. It is truncated if longer than 8 characters. (This variable is valid only in the pending requests form, `pendingmsg.form`. It is ignored in the other forms.)
%%dn%%	The subject's distinguished name. (This variable is valid in all the forms except the pending requests form, `pendingmsg.form`.)
%%printcert%%	A renewed certificate in Base64-encoded format. (This variable is valid only in the renewed certificate form, `renewcertmsg.form`. It is ignored in the other forms.)
%%modreqlist%%	A list of pending approval requests (or a single request) that are modified by an administrator. (This variable is valid only in the pending modified message form, `pendingmsg2.form`. It is ignored in the other forms.)
%%notafter%%	The certificate expiration date and time in local time in the format `YYYY/MM/DD HH:MM:SS`. (This variable is valid only in the `expiring.form`. It is ignored in the other forms.)
%%pendreqlist%%	A list of pending approval requests (or a single request). Each request contains the transaction ID followed by the corresponding requestor. Each request should be on a line of its own. (This variable is valid only in the pending message form, `pendingmsg.form`. It is ignored in the other forms.)

%%quicklink%%	A link to a certificate that is ready for pickup. It contains the transaction ID, the "&" character, and the string of the template name or alias, with escaped characters. The following string (broken into two lines so that it fits in the column) is an example:
	 `TransactionId=1j86b0wokkoQ2SHV%2B%2B%2B%2B`
	 `%2B%2B%2B%2B&Template=PKI+Browser+Certificate`
	 The template name or alias is located from the appldata field of the certificate from the issued certificate list (ICL). The appldata field corresponds to the value of the NICKNAME directive in the template. If PKI Services cannot determine the template name or alias, the `%%quicklink%%` variable is an empty string, and a warning level message IKYC056I is logged.

%%recoverylink%%	Part of the link to a certificate that can be recovered. It contains the serial number, the "&" character, and the KeyId of the recovery certificate. This part of the link is at the end of the line, which is appended to the URL link. The entire line is repeated for each entry in the list of certificates that can be recovered.

Table 55. Descriptions of variables for forms (continued)

Variable	Description
%%recoverylist%%	The list of certificates that meet the criteria for recovery; that is, the email address for the certificate matches the email address of the user requesting the recovery of the certificate, the password used for the original certificate request matches the password provided by the user, and PKI Services created the keys for the certificate. The URL for each recovered certificate is on a line by itself.
%%rejectreason%%	The reason for the rejection of a certificate request. (This variable is valid for the reject form only.)
%%requestor%%	The requestor of the certificate.
%%transactionid%%	The transaction ID (CertId) returned. (This variable is valid for the ready and reject forms only. It is ignored in the other forms.)

Table 56 on page 322 summarizes which substitution variables are supported by which forms:

Table 56. Summary of substitution variables in forms

Substitution variable	readymsg	rejectmsg	expiringmsg	pendingmsg	pendingmsg2	renewcertmsg	recoverymsg
%%cadomain%%	(ignored)	(ignored)	(ignored)	X	(ignored)	(ignored)	(ignored)
%%dn%%	X	X	X	(ignored)	(ignored)	X	(ignored)
%%printcert%%	(ignored)	(ignored)	(ignored)	(ignored)	(ignored)	X	(ignored)
%%modreqlist%%	(ignored)	(ignored)	(ignored)	(ignored)	X	(ignored)	(ignored)
%%notafter%%	(ignored)	(ignored)	X	(ignored)	(ignored)	(ignored)	(ignored)
%%pendreqlist%%	(ignored)	(ignored)	(ignored)	X	(ignored)	(ignored)	(ignored)
%%quicklink%%	X	(ignored)	(ignored)	(ignored)	(ignored)	(ignored)	(ignored)
%%recoverylink%%	(ignored)	(ignored)	(ignored)	(ignored)	(ignored)	(ignored)	X
%%recoverylist%%	(ignored)	(ignored)	(ignored)	(ignored)	(ignored)	(ignored)	X
%%rejectreason%%	(ignored)	X	(ignored)	(ignored)	(ignored)	(ignored)	(ignored)
%%requestor%%	X	X	X	(ignored)	(ignored)	X	X
%%transactionid%%	X	X	(ignored)	(ignored)	(ignored)	(ignored)	(ignored)

Steps for customizing email notification forms

Perform the following steps to customize the forms:

1. Make sure the forms you want to use (`readymsg.form, rejectmsg.form, expiringmsg.form, renewcertmsg.form, pendingmsg.form`, and `recoverymsg.form`) are present in the runtime directory. (By default, the runtime directory is `/etc/pkiserv/`. For information about copying files, see Step "3" on page 70.)

   ____

2. Update the form. At minimum:

   a. Specify your company (instead of `dime-o-cert`) in the `From:` line and in the first line of the main paragraph

   b. If appropriate, update the subject.

Note: There must be a blank line between the subject and the body of the note.

c. If you are updating a ready, expiring, or recovery form, change the URL in the main paragraph to customize it for your company. The sample forms have URLs for both REXX CGI exec and JavaServer page (JSP) use. Modify the URL that applies for the method you are using to implement the web application, and remove the URL that does not apply. Remove the comment lines preceding each URL.

   For example, if you are modifying the ready form, and you are implementing the web application using JSPs, change:

```
Please visit:
(edit the following link if using the REXX CGI Web Application)
https://www.dimeocert.com/Customers/ssl-cgi-bin/caretrieve.rexx?%%quicklink%%
(edit the following link if using the JSP WebSphere Application)
https://www.dimeocert.com:9080/PKIServ_Web/Customers/certretrieve.jsp?%
%quicklink%%
to retrieve your certificate.
```

   to

```
Please visit:
https://www.dimeocert.com:9080/PKIServ_Web/Customers/certretrieve.jsp?%
%quicklink%%
to retrieve your certificate.
```

   and modify the URL for your company.

d. If you are updating a reject form, change the telephone number in the main paragraph to customize it for your company.

Make any other needed changes. (You can use variables in the body of the form, but you cannot include `%%transactionid%%` in the expiring form or `%%notafter%%` in the ready or reject form.)

____

3. Save the file.

____

Setting up automatic renewal of certificates

You can optionally set up PKI Services to automatically renew certificates when they approach their expiration date, and email the new certificates to their owners. This option is controlled on a template basis.

Steps for setting up automatic certificate renewal

Perform the following steps to set up automatic certificate renewal.

Before you begin

You need to decide for which certificate templates you want to set up automatic certificate renewal. For a description of the templates provided by PKI Services, see "Supported certificate types" on page 7.

Procedure

1. In the CertPolicy section of the pkiserv.config configuration file, set the field ExpireWarningTime to specify how soon (in days or weeks) before a certificate expires to renew it and send the renewed certificate to its owner. For example, to automatically renew certificates two weeks before they expire:

```
ExpireWarningTime=2w
```

____

2. Set up the renewed certificate email notification form.

a) Copy the sample renewed certificate notification form, renewcertmsg.form, from the samples directory to the runtime directory. For more information, see "Steps for copying files" on page 69.

b) Customize the renewed certificate notification form with your company's information. For more information, see "Customizing email notifications sent to users" on page 317.

c) In the General section of the pkiserv.config configuration file, set the field RenewCertForm to indicate the file that contains the renewed certificate notification form. For example:

```
RenewCertForm=/etc/pkiserv/renewcertmsg.form
```

____

3. If you are implementing the web application using REX CGI execs, in each template for which you want certificates to be automatically renewed, insert the AUTORENEW tag immediately following the NICKNAME tag, if it is not already there, and set it to Y. For example:

```
<TEMPLATE NAME=1-Year PKI SSL Browser Certificate>
<TEMPLATE NAME=PKI Browser Certificate>
<NICKNAME=1YBSSL>
<AUTORENEW=Y>
```

If you are implementing the web application using JavaServer pages (JSPs), for each certificate request template for which you want certificates to be automatically renewed, include the tag `<tns:AutoRenew>Y</tns:AutoRenew>`. For example:

```
<tns:certreq_template>
<tns:certname>1-Year SAF Browser Certificate</tns:certname>
<tns:certtype>SAF Browser Certificate</tns:certtype>
<tns:AutoRenew>Y</tns:AutoRenew>
⋮
```

____

4. For each certificate type that you want to be automatically renewed, except the PKI generated key certificate, make NotifyEmail a required field.

To do this if you are implementing the web application using REX CGI execs, in each template for which you want certificates to be automatically renewed (except the PKI generated key certificate), remove the string `(optional)` following the NotifyEmail tag, if it is specified. For example, change

```
 %%NotifyEmail (optional)%%
```

to

```
 %%NotifyEmail%%
```

If you are implementing the web application using JavaServer pages (JSPs), for each certificate request template for which you want certificates to be automatically renewed (except the PKI generated key certificate), remove the string `optional="true"` following the NotifyEmail tag, if it is specified. For example, change:

```
<tns:NotifyEmail optional="true" />
```

to

```
<tns:NotifyEmail />
```

Note: For a PKI generated key certificate, the requestor name is an email address and overrides the NotifyEmail value if specified.

____

Results

When you are done, you have set up automatic certificate renewal.

Setting up synchronous certificate requests

You can optionally set up PKI Services to request certificates synchronously. When the requestor submits a request, and gets back a transaction ID, the certificate has been generated. The requester can pick up the certificate immediately. This option is controlled on a template basis.

If you are implementing the web application using REX CGI execs, in each template for which you want certificates to be requested synchronously, insert the SYNCRHRONOUS=Y tag before the CONTENT tag.

For example:

```
#<AUTORENEW=Y>
<SYNCHRONOUS=Y>
<CONTENT>
```

If you are implementing the web application using JavaServer pages (JSPs), for each certificate request template for which you want certificates to be requested synchronously, include the following tags `<tns:Synchronous>Y</tns:Synchronous>` after `<tns:AutoApprove>Y</tns:AuthoApprove>`.

For example:

```

<tns:certname>2-Year PKI Browser Certificate For Authenticating To z/OS</tns:certname>
<tns:certtype>PKI Browser Certificate</tns:certtype>
<tns:request_authtype>zAuthRunAsSurrogate</tns:request_authtype>
<tns:retrieve_authtype>zAuthRunAsSurrogate</tns:retrieve_authtype>
<tns:AutoApprove>Y</tns:AutoApprove>
<tns:Synchronous>Y</tns:Synchronous>
```

Setting up PKI Services to generate keys for certificate requests

There are two ways to generate the key pair (public key and private key) for a certificate request:

- The requestor can generate the key pair and send the public key to PKI Services with the request. In this case, PKI Services has no knowledge of the private key, and cannot recover it if the requestor loses it.
- The requestor can ask PKI Services to generate the key pair. PKI Services uses the PKCS #11 API provided by ICSF to generate the key pair and store it in the token data set (TKDS). The requestor's email address is used as the requestor name. The certificate and private key are packaged in PKCS #12 format and the requestor is sent a link from email to retrieve the package. In this case PKI Services can recover the certificate package if the requestor needs it.

Before PKI Services can generate key pairs for certificates, you must do some setup. For example, you must set the `TokenName` keyword in the `pkiserv.conf` configuration file.

Requirement: The key generation capability requires hardware that supports the PKCS #11 CKM_RSA_PKCS_KEY_PAIR_GEN mechanism. For information about which hardware supports this mechanism, see *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*.

PKI Services can generate both secure keys and clear keys. The sensitive key material of a *secure key* is wrapped under a master key. A *clear key* is not encrypted. You control whether PKI services generates secure keys or clear keys by setting the `SecureKey` keyword in the `pkiserv.conf` configuration file. You can restrict the generation of clear keys by defining a profile protecting the resource `CLEARKEY.`*`token_name`* in the CRYPTOZ class.

Rules:

- The `SecureKey` keyword is ignored if the `TokenName` keyword is not specified.
- If the `SecureKey` keyword is set to T, PKI Services generates secure keys. (The CKA_IBM_SECURE attribute is set to TRUE.)
- If the `SecureKey` keyword is set to F or is not specified, PKI Services generates clear keys if profiles in the CRYPTOZ class that protect the CLEARKEY function allow clear key generation. If CLEARKEY

profiles do not allow clear key generation, PKI services generates a secure key. For example, the following RACF command prevents clear key generation on the token named PKISRVD.PKITOKEN:

```
RDEF CRYPTOZ CLEARKEY.PKISRVD.PKITOKEN UACC(NONE)
```

Steps for setting up PKI Services to generate keys for certificate requests

Perform the following steps to set up key generation for certificate requests.

Before you begin

- You need to know whether the ICSF token data set (TKDS) has already been set up.
- You need to know whether you want PKI Services to generate secure or clear keys.
- You need to know whether you want to restrict the use of clear keys.

Procedure

1. If the ICSF token data set (TKDS) has not already been set up, ask the ICSF programmer to set it up. (For information about the TKDS, see *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*.) The TKDS must be set up before PKI Services starts, so if necessary stop and restart PKI Services after the TKDS is set up. (For information about stopping and restarting PKI Services, see Chapter 10, “Starting and stopping PKI Services,” on page 129.

   ____
2. Edit the **SAF** section of the PKI Services configuration file, `pkiserv.conf`, and verify that the `TokenName` parameter is specified. If it is not, choose a name for the token in the TKDS that PKI Services uses for storing the key pairs that it generates, and set `TokenName` to the name you choose.

   Rules: A token name must follow these rules:

   - Up to 32 characters in length
   - Permitted characters are:
     - Alphanumeric
     - National: "@" (X'5B'), "#" (X'7B'), or "$" (X'7C')
     - Period: "." (X'4B')
   - The first character must be alphabetic or national
   - Lowercase letters can be used, but are folded to uppercase
   - The IBM1047 code page is assumed

   ____
3. Edit the **SAF** section of the PKI Services configuration file, `pkiserv.conf` and set the `SecureKey` parameter to T if you want PKI Services to generate secure keys, or to F if you want PKI Services to generate clear keys.

   ____
4. If you want to restrict the generation of clear keys, have the security administrator create a profile in the CRYPTOZ class to do so.
   For example, if you set `TokenName` to PKISRVD.PKITOKEN, to restrict PKI Services from generating clear keys the security administrator issues the command:

   ```
   RDEF CRYPTOZ CLEARKEY.PKISRVD.PKITOKEN UACC(NONE)
   ```
5. Edit the **General** section of the PKI Services configuration file, `pkiserv.conf`, and verify that the `ReadyMessageForm` parameter is specified. If it is not:

   - Copy the ready message form from the samples directory to the runtime directory. Follow the instructions in “Steps for copying files” on page 69.

- Update the `ReadyMessageForm` parameter to specify the full path name or data set name of the ready message form.
- Customize the ready message form. Follow the instructions in "Customizing email notifications sent to users" on page 317.

____

6. Set up the email form that is sent if a user requests that PKI Services recover a certificate for which PKI Services generated the keys. The form contains a list of certificates that can be recovered. Edit the **General** section of the PKI Services configuration file, pkiserv.conf and verify that the `RecoverForm` parameter is specified. If it is not:

   - Copy the recovery message form from the samples directory to the runtime directory. Follow the instructions in "Steps for copying files" on page 69.
   - Update the `RecoverForm` parameter to specify the full path name or data set name of the recovery message form.
   - Customize the recovery message form. Follow the instructions in "Customizing email notifications sent to users" on page 317.

____

7. If you want expired certificates whose keys were generated by PKI Services to be deleted from the ICL automatically after a certain time period, edit the **ObjectStore** section of the PKI Services configuration file, `pkiserv.conf` and update the `RemoveExpiredCertsAndKeys` parameter to specify the time period after which the expired certificates should be deleted.

____

8. The RACF administrator must give the PKI Services daemon the authorization it needs to use the PKCS #11 APIs. The following RACF commands set up the required authorization in the CRYPTOZ class. (By default the daemon user ID is PKISERVD, but you might be using a different user ID. Check the `daemon` variable in Table 19 on page 55 if you are not sure what your daemon user ID is.)

```
SETROPTS CLASSACT(CRYPTOZ) GENERIC(CRYPTOZ) RACLIST(CRYPTOZ)
RDEFINE CRYPTOZ SO.daemon_id.* UACC(NONE)
RDEFINE CRYPTOZ USER.daemon_id.* UACC(NONE)
PERMIT SO.daemon_id.* CLASS(CRYPTOZ) ID(daemon_id) ACC(UPDATE)
PERMIT USER.daemon_id.* CLASS(CRYPTOZ) ID(daemon_id) ACC(CONTROL)
SETROPTS RACLIST(CRYPTOZ) REFRESH
```

   Note: These commands are included in the IKYSETUP REXX exec. If you have another reason to rerun IKYSETUP, you can update the exec to set up the daemon user ID's authorization in the CRYPTOZ class at the same time. If you do not have another reason to rerun IKYSETUP, the RACF administrator can issue the commands manually.

____

9. (Optional) Because PKI Services stores certificates for which it generates the keys in the TKDS, it can recover those certificates from the TKDS. To recover a certificate, a user must provide the passphrase that was entered when the certificate was originally requested. If the user has forgotten the passphrase, you can use the PKI Services exit to allow the user to recover the passphrase by responding to security questions. For more information, see "Scenario 4: Allow users to recover a PKI generated key certificate when the passphrase is lost" on page 361. Decide whether you want to implement passphrase recovery, and if so, write exit code to implement the function.
10. (Optional) When PKI Services generates the keys for certificate requests, it returns both the certificate and the private key in PKCS#12 format when retrieved by the requester. By default, the PKCS#12 contains the CA certificate that signed the requested certificate. The contents of the PKCS#12 may be tailored to contain only the issued certificate with private key, the issued certificate with private key, and the CA certificate that signed the issued certificate, or the issued certificate with private key and the complete CA signing chain. Use the `PKCS12Content` keyword in the `pkiserv.conf` file to specify the PKCS#12 content. When `PKCS12Content` is set to either **I** or **E**, no further configuration is required. However, when `PKCS12Content=C` is set, if the PKI CA certificate is not a root CA (self-signed), each of the CA certificates in the signing chain must be connected to the PKI Services key ring with CERTAUTH usage. For example, If the PKI Services CA

certificate is signed by a CERTAUTH certificate with a Label of "Dime-o-Cert Root CA", and the `pkiserv.conf` file has Keyring=PKISRVD/CAring in the [SAF] section, the following RACF command is issued to connect the root CA certificate to the PKI Services key ring:

```
RACDCERT ID(PKISRVD) CONNECT(CERTAUTH LABEL('Dime-o-Cert Root CA') ring(CAring))
```

Results

When you are done, PKI Services can generate key pairs for certificate requests if asked to do so.

Adding custom extensions to certificates

PKI Services supports the following standard certificate extensions:

- Authority Key Identifier
- Subject Key Identifier
- Key Usage
- Extended Key Usage
- Subject Alternate Name
- BasicConstraints
- CRL Distribution Point
- HostIDMapping
- Authority Information Access
- Certificate Policies

In addition, PKI Services supports custom extensions. You can include any extension in your certificates that is in the form:

```
Extension  ::=  SEQUENCE  {
     extnID      OBJECT IDENTIFIER,
     critical    BOOLEAN DEFAULT FALSE,
     extnValue   OCTET STRING  }
```

PKI Services provides a CustomExt INSERT and a CustomExt JSP that you can use to add custom extensions to your certificate templates. The *n*-year PKI browser certificate template demonstrates the use of custom extensions.

Restriction: Do not add a critical extension to a certificate template for which PKI Services generates the keys.

Steps for adding a custom extension to a certificate template if you are using REXX CGI execs

Procedure

1. Copy the CustomExt INSERT from `pkiserv.tmpl`.

   ____

2. Rename the copy of the CustomExt INSERT. The name is the string that is formed by concatenating the following values:

   a) The string "CustomExt".

   b) The OID. You are responsible for ensuring that you use a registered OID, PKI Services does not check this.

   c) The critical flag - C for a critical extension, N for a non-critical extension.

   d) The value type. Supported types are:

- INT (integer in a printable hexadecimal format)
- IA5 (IA5 string)
- PRT (printable string)
- BMP (BMP string)
- OCT (Octet string)
- UTF (UTF 8 string)

____

3. Customize the INSERT to contain any inputs that you want, and to have the corresponding JavaScript code manipulate and verify the inputs.

   ____

4. Add the customized INSERT to the certificate template that you want to use it in.

   ____

5. Update the CONTENT subsection of the certificate template to add new input fields.

   ____

Results

When you are done, you have added a customized extension to a certificate template using REXX CGI execs.

Steps for adding a custom extension to a certificate template if you are using JSPs

Procedure

1. Copy the CustomExt JSP from the `mod.inc` directory, and the certificate template file, `pkitmpl.xml`, by default in the `/etc/pkiserv/` directory.

   ____

2. Rename the copy of the CustomExt JSP. The name is the string formed by concatenating the following values:

   a) The string "CustomExt".

   b) The OID. You are responsible for ensuring that you use a registered OID, PKI Services does not check this.

   c) The critical flag - `C` for a critical extension, `N` for a non-critical extension.

   d) The value type. Supported types are:

      - INT (integer in a printable hexadecimal format)
      - IA5 (IA5 string)
      - PRT (printable string)
      - BMP (BMP string)
      - OCT (Octet string)
      - UTF (UTF 8 string)

   ____

3. Customize the certificate template file, `pkitmpl.xml` to contain any inputs that you want, and customize the JSP file to manipulate and verify the inputs.

   ____

4. Use the TemplateTool utility to validate your updated XML template file. When you have completed your updates and successfully validated them, use the TemplateTool utility to create an updated CGI template file, `pkixgen.tmpl`

____

5. Update the EAR file with the modified JSP file, and deploy the EAR file to a WebSphere Application Server. For information on how to do this, see "Customizing the PKI Services web application" on page 257.

   ____

Results

When you are done, you have added a customized extension to a certificate template using JSPs.

Forming the CustomExt value for CertPlist for the R_PKIServ callable service

The INSERT or JSP code that you write for a custom extension must be able to call the R_PKIServ callable service with a properly formed `CustomExt` value for the `CertPlist` parameter list. The `CustomExt` `CertPlist` entry is a comma-separated 4-part string with a maximum length of 1024 bytes:

1. The first part is the OID of the extension.
2. The second part is the critical flag:
   - C or c indicates critical
   - N or n indicates non-critical
3. The third part is the encode type:
   - INT indicates integer
   - IA5 indicates IA5 string
   - PRT indicates printable string
   - BMP indicates BMP string
   - OCT indicates octet string
   - UTF indicates UTF string
4. The fourth part is the value.

For example, given the following ASN.1 notation for an x.509 extension:

```
Extension :: = SEQUENCE {
    extnID    OBJECT IDENTIFIER,
    critical    BOOLEAN DEFAULT FALSE,
    extnValue    OCTET STRING
}
```

and the following definition of an ASN.1 custom extension:

```
SEQUENCE {
. OBJECT IDENTIFIER '1 2 3 4'
. OCTET STRING, encapsulates {
. . INTEGER 240
. . }
. }
```

the `CustomExt` value to produce this would be:

```
1.2.3.4,N,INT,F0
```

Rules: The OID value must follow these rules:

- Its length must not exceed 64 characters.
- It must be comprised of decimal digits and the dot (period) character).
- It cannot start or end with a dot (period) character and cannot contain adjacent dots.
- The first integer value must be 0, 1, or 2. If the first integer is 0 or 1, the second integer must not be greater than 39.

- It must contain a minimum of 3 dot-separated segments (for example 1.2.3).
- The value of the first segment must be 0, 1, or 2. If the first integer is 0 or 1, the second integer must not be greater than 39.
- The largest integer value for all segments after the second segment is the largest 31-bit integer value: 2 147 483 647.

Rules: The critical flag must follow these rules:

- Only mark an extension critical if all applications that use the certificate know and understand the extension. Applications that do not know and understand an extension that is marked critical are required to deny the use of the certificate.
- Do not mark an extension critical if PKI Services creates the public and private keys for the certificate.

Rules: The encoding types and values must follow these rules:

- The INT encoding type indicates that the extension value encapsulated in the extension value's octet string is a primitive `integer` type.
- Because the INTEGER type in the ASN.1 specification is not limited to fit in a 4-byte construct, the value specified for INT is a hexadecimal string rather than decimal. If the specified hexadecimal string for an INTEGER type contains an odd number of characters, the high-order bit of the first character is propagated in the encoded value. For example, if the `CustomExt` value is `12.3.4,N,INT,800`, because there are only 3 characters in 800, and the high order of the first character is on, the encoded value is F800 in hexadecimal, or -2048 decimal.
- The PRT encoding type indicates that the extension value encapsulated in the extension value's octet string is a primitive `printableString` type. The characters specified in the `CustomExt` value must conform to the `PrintableString` character set, which is a subset of the 7-bit ASCII or IA5 character set.
- The IA5 encoding type indicates that the extension value encapsulated in the extension value's octet string is a primitive `IA5String` type. The characters specified in the `CustomExt` value must conform to the IA5String character set. Control characters that part of the IA5 character set are not allowed.
- THE UTF encoding type indicates that the extension value encapsulated in the extension value's octet string is a primitive `UTF8String` type. The characters specified in the `CustomExt` value must conform to the Basic Latin and Latin-1 characters except for control characters. To encode a `UTF8String` extension value with characters that fall outside the range supported, you must use the OCT encoding type.
- The BMP encoding type indicates that the extension value encapsulated in the extension value's octet string is a primitive `BMPString` type. The characters specified in the `CustomExt` value must conform to the Basic Latin and Latin-1 characters except for control characters. To encode a `BMPString` extension value with characters that fall outside the range supported, you must use the OCT encoding type.
- The OCT encoding type indicates that the extension value specified in the `CustomExt` value is the extension's octet string value. You must specify an even number of printable hexadecimal characters, and the `OctetString` value should not include the tag and length bytes for the `OctetString`.

Example: You want to build a custom extension whose `extnID` is 1.2.3.4, is not marked critical, and has an `extnValue` composed of a SEQUENCE containing an INTEGER value of 1024 (decimal) followed by a UTF-8 string with the value "AB£¬¥YZ". The ASN.1 definition is:

```
SEQUENCE {
. OBJECT IDENTIFIER '1 2 3 4'
. OCTET STRING, encapsulates {
. . SEQUENCE {
. . . INTEGER 1024
. . . UTF8String 'AB£¬¥YZ'
. . . }
. . }
. }
```

Use the following value for `CustomExt`:

```
CustomExt=1.2.3.4,N,OCT,3010020204000C0A4142C2A3C2ACC2A5595A
```

For the OCT value:

```
3010 = SEQUENCE(0x30) of length 0x10
02020400 = INTEGER(0x02) of length 0x02, value 0x0400 = 1024 decimal
0C0A4142C2A3C2ACC2A5595A = UTF8String(0x0C) of length 0x0A,
    value = 41(A) 42(B) C2A3(£) C2AC(¬) C2A5(¥) 59(Y) 5A(Z)
```

Chapter 15. Enabling Simple Certificate Enrollment Protocol (SCEP)

The Simple Certificate Enrollment Protocol (SCEP) allows you to securely issue certificates to large numbers of network devices using an automatic enrollment technique. The network devices, usually IPSEC devices such as Cisco routers, must be SCEP-enabled and preregistered (to your CA domain) before they can successfully request certificates from you. To request a certificate, the preregistered SCEP client sends a message (the certificate request) to your CA using the HTTP protocol. (The message is a PKCS #10 request enveloped in a signed PKCS #7 structure.)

You can configure PKI Services to respond automatically to some (or all) SCEP certificate requests, or to submit some (or all) SCEP certificate requests to the PKI administrator for approval or rejection. When you enable automatic enrollment, certificate requests can be automatically approved and synchronously fulfilled, based on the requestor's knowledge of a predetermined secret, the challenge passphrase.

Overview of SCEP requester enhancement

In releases prior to Version 2 Release 3, Simple Certificate Enrollment Protocol (SCEP), used the alternate index on the requestor field of the SCEP request for searching purposes. The SCEP request contains a transaction ID, which is a hash of its public key. The leftmost 32-bytes of the SCEP ID are used as the requestor field in the certificate request. The client name of the preregistration record is not saved. In Version 2 Release 3, two header fields are added.

- Header field to store the version of the object store and ICL.
- Header field to save the SCEP transaction ID, instead of using the requestor field.

The existing requestor field is used to save the client name. As a result, conversions of the existing object store and ICL must be completed by using the `vsamconv` utility for VSAM data sets or the `db2conv` utility for Db2.

Note: If you are setting up a new PKI Services instance in Version 2 Release 3, specify `DBVersion=1` in `pkiserv.conf`.

Steps for converting existing VSAM object store and ICL to the new format

1. Run IKYCVSV1 to create the new VSAM data sets with the additional alternate index for the SCEP transaction ID field.
2. Stop PKI Services.
3. Run the conversion utility `vsamconv`, using the existing ICL and object store VSAM data sets as the source data sets and the VSAM data sets created in Step "1" on page 333 as the target data sets.
4. Update `pkiserv.conf` to specify the DBVersion to 1 and to use the object store and ICL data sets created in Step "1" on page 333.
5. Start PKI Services.

Steps for converting existing Db2 object store and ICL to the new format

1. Run IKYCDBV1 to create the new Db2 ObjectStore and ICL tables.
2. Run IKYSBIND by using the `version 1` DBRM, IKYPDBR1, to build the new package with the new name.
3. Stop PKI Services.
4. Run the conversion utility `db2conv`, using the previously existing package as the source package and the newly create package from Step 2 as the target package.

5. Update `pkiserv.conf` to specify the DBVersion to 1 and change the DBPackage to match the name of the new package that is created in Step 2.
6. Start PKI Services.

Enabling Simple Certificate Enrollment Protocol (SCEP)

The Simple Certificate Enrollment Protocol (SCEP) allows you to securely issue certificates to large numbers of network devices using an automatic enrollment technique. The network devices, usually IPSEC devices such as Cisco routers, must be SCEP-enabled and preregistered (to your CA domain) before they can successfully request certificates from you. To request a certificate, the preregistered SCEP client sends a message (the certificate request) to your CA using the HTTP protocol. (The message is a PKCS #10 request enveloped in a signed PKCS #7 structure.)

You can configure PKI Services to respond automatically to some (or all) SCEP certificate requests, or to submit some (or all) SCEP certificate requests to the PKI administrator for approval or rejection. When you enable automatic enrollment, certificate requests can be automatically approved and synchronously fulfilled, based on the requestor's knowledge of a predetermined secret, the challenge passphrase.

Overview of SCEP preregistration

To request certificates using SCEP, a SCEP requestor must be *preregistered* to PKI Services, your CA. You can preregister SCEP clients in batches using the `pkiprereg` utility (see "Using the pkiprereg utility" on page 437) or the PKI administrators can preregister individual SCEP clients (one client at a time) using the end-user web pages.

When PKI administrators preregister a SCEP client, they do so by using the end-user web page for requesting a certificate and selecting the SCEP (preregistration) certificate template called `5-Year SCEP Certificate - Preregistration`. (See "Steps for preregistering an SCEP or EST client" on page 403.) The PKI administrator fills out the request form by specifying the device or client name of the SCEP client, a passphrase for client authentication, and additional (optional) subject name and alternate name information. You can customize the **<CONSTANT>** section of the SCEP (preregistration) certificate template to supply the additional optional information.

When a PKI administrator submits the form for a SCEP (preregistration) certificate request, PKI Services creates a preregistration record, not an actual certificate request, in the VSAM ObjectStore data set (request database). The client name is translated to lowercase characters, truncated to 32 characters if longer, and saved as the `Transaction ID` to support searching of the ObjectStore. (Each preregistration record must have a client name that is unique in the first 32 characters, regardless of uppercase or lowercase.)

The preregistration record contains the template nickname, passphrase, and additional (optional) subject name and alternate name values. Any other information (unrelated to the subject name or alternate name) specified on the request form is ignored.

When you customize the **<CONSTANT>** section of the SCEP template to supply additional (optional) values for the following variables, those values are not saved in the preregistration record. However, those values are processed when the preregistered client then requests a certificate.

- AuthInfoAcc
- CertPolicies
- Critical
- ExtKeyUsage (not typically used in a SCEP request)
- KeyUsage (not typically used in a SCEP request)
- NotAfter
- NotBefore

Overview of certificate request processing for preregistered SCEP clients

Following preregistration, when the preregistered SCEP client requests a certificate (sends a SCEP request), PKI Services searches for a preregistration record matching the client name. If found, PKI Services compares the values in the request to the challenge password and any subject name or alternate name information specified by the PKI administrator or supplied in the **<CONSTANT>** template section. (If not found, the SCEP request is automatically rejected.)

Based on the comparison of values in the request with those in the preregistration record, PKI Services considers the request to be in one of the following states:

Authenticated
When the challenge password matches and all other preregistered values are included in the request

Semiauthenticated
When the challenge password matches but some other preregistered values are missing from the request

Unauthenticated
When the challenge password does not match or is missing.

Depending on how you customize the variables in the SCEP (preregistration) certificate template, a certificate request from an `Authenticated` SCEP client is either automatically approved and fulfilled synchronously or it is queued for administrator approval. Likewise, a certificate request from an `Unauthenticated` or `Semiauthenticated` SCEP client is either queued for administrator approval or it is automatically rejected.

Variables used in the <PREREGISTER> section

These are the valid variables that you can customize in the **<PREREGISTER>** section of the `5-Year SCEP Certificate - Preregistration` template. Some variables *must* be present in your **<PREREGISTER>** section and they are labeled as *required* in the following list.

AuthenticatedClient *(required)*
Specifies which action PKI Services takes when an authenticated SCEP client submits a certificate request for the first time. Valid values are:

AutoApprove *(default)*
Automatically approves certificate requests from authenticated first-time SCEP clients and automatically creates their certificates.

AdminApprove
Submits certificate requests from authenticated first-time SCEP clients to your PKI administrator for verification and approval. The ADMINNUM tag, when present, indicates that multiple approvals are required.

SemiauthenticatedClient *(required)*
Specifies which action PKI Services takes when a semiauthenticated SCEP client submits a certificate request for the first time. Valid values are:

AdminApprove *(default)*
Submits certificate requests from semiauthenticated first-time SCEP clients to your PKI administrator for verification and approval. The ADMINNUM tag, when present, indicates that multiple approvals are required.

Reject
Automatically rejects certificate requests from semiauthenticated first-time SCEP clients.

UnauthenticatedClient *(required)*
Specifies which action PKI Services takes when an unauthenticated SCEP client submits a certificate request for the first time. Valid values are:

AdminApprove
Submits certificate requests from unauthenticated first-time SCEP clients to your PKI administrator for verification and approval. The ADMINNUM tag, when present, indicates that multiple approvals are required.

Reject *(default)*
Automatically rejects certificate requests from unauthenticated first-time SCEP clients.

SubsequentRequest *(optional)*
Specifies which action PKI Services takes when a previously approved SCEP client submits an additional certificate request. If not set, PKI Services uses the **AuthenticatedClient** value. Valid values are:

AutoApprove *(default)*
Automatically approves certificate requests from previously approved SCEP clients and automatically creates their certificates.

AdminApprove
Submits certificate requests from previously approved SCEP clients to your PKI administrator for verification and approval. The ADMINNUM tag, when present, indicates that multiple approvals are required.

Reject
Automatically rejects SCEP requests from previously approved clients.

RenewalRequest *(optional)*
Specifies which action PKI Services takes when a previously approved SCEP client submits a certificate renewal request. If not set, PKI Services uses the **AuthenticatedClient** value. Valid values are:

AutoApprove *(default)*
Automatically approves certificate renewal requests from previously approved SCEP clients and automatically creates their certificates.

AdminApprove
Submits certificate renewal requests from previously approved SCEP clients to your PKI administrator for verification and approval. The ADMINNUM tag, when present, indicates that multiple approvals are required.

Reject
Automatically rejects certificate renewal requests from previously approved SCEP clients.

Tags used in the <PREREGISTER> section

The following tags are allowed in the **<PREREGISTER>** section of a template. Use of these tags is optional unless otherwise specified.

<ADMINNUM= *value*>
This optional tag indicates the number of PKI Services administrators that must approve certificate requests queued for approval before a certificate can be issued. If this tag is present, any variables that are assigned the value of **AdminApprove** requires the number of approvals that are specified by this tag. This tag has the form <**ADMINNUM**= *value*>, where *value* can be a numeric value from 1 to 32. The tag has the following meanings:

- By default, queued requests require approval by one PKI Services administrator. If the **ADMINNUM** tag is not present, all queued requests require approval by one PKI Services administrator.
- If the **ADMINNUM** tag does not occur within the **PREREGISTER** subsection, PKI Services operates as if the tag is not present.
- If the **ADMINNUM** value is greater than 32, a value of 32 is used.
- If the **ADMINNUM** value is less than one or is a non-numeric value, a value of 1 is used.

Note: A request created from this template remains Pending Approval state until the required number of individual administrative approvals is made for the request, at which time the request changes to Approved state. If an administrator issues an Approve with Modifications on a request that is in Pending Approval state, any previously made approvals are nullified, and the number of approvals that are made for the request is reset to 1.

Checking certificate fingerprints

There are two instances when the PKI administrator checks certificate *fingerprints* (the SHA1, MD5, SHA256, and SHA512 hashes) in support of certificate request processing for SCEP clients.

- Preregistered SCEP clients who request certificates from this CA domain must download the correct PKI Services CA certificate to their workstations before they issue their certificate requests. After the download, the client can use the SCEP client software to display the fingerprints of the downloaded CA certificate and then confirm with the PKI administrator of the CA domain that it is the correct CA certificate.

  To match CA certificate fingerprints with a SCEP client, the PKI administrator can display the fingerprints of the CA certificate for this domain by issuing the following MODIFY (or **F**) console command:

  ```
  F PKISERVD,DISPLAY
  ```

  The result of this command is information message IKYP025I. **Sample output:**

  ```
  SY1  IKYP025I PKI SERVICES SETTINGS:
    CA DOMAIN NAME: Customers
    SUBCOMPONENT               MESSAGE LEVEL
       POLICY                  WARNING MESSAGES AND HIGHER
       LDAP                    ERROR MESSAGES AND HIGHER
       SAF                     WARNING MESSAGES AND HIGHER
       DB                      INFORMATIONAL MESSAGES AND HIGHER
       CORE                    WARNING MESSAGES AND HIGHER
       PKID                    VERBOSE DIAGNOSTIC MESSAGES AND HIGHER
       TPOLICY                 WARNING MESSAGES AND HIGHER
    MESSAGE LOGGING SETTING: STDOUT_LOGGING
    CONFIGURATION FILE IN USE:
  /etc/pkiserv/pkiserv.conf
    TEMPLATE FILE IN USE:
  /etc/pkiserv/pkiserv.tmpl
    CA CERTIFICATE FINGERPRINTS:
     SHA1:   25:25:EE:1E:B1:3B:5D:92:E5:3B:74:0D:0C:F3:C3:93:F5:9A:78:ED
     SHA256: F6:C7:C4:B2:4F:99:1F:01:04:C2:AC:07:B3:8B:4A:69:
             78:6E:46:0C:95:E9:93:D9:BF:0A:F1:C1:49:C8:AD:2E
     SHA512: 84:A3:E8:2E:67:2E:F2:16:30:A8:ED:92:32:1D:F5:5F:
             14:58:63:B9:2B:B6:4F:B4:60:59:32:8C:69:8B:B7:CA:
             5C:AD:7D:C8:EF:6F:F9:20:5E:1B:B7:42:79:1C:53:1C:
             11:2D:5E:DD:25:A9:32:3A:3C:D2:CD:14:DB:B6:CA:65
    FIPS LEVEL: FIPS 140-2
    STATUS: OPERATIONAL
  ```

  Note: The MD5 fingerprint is only displayed if FIPS mode is off.

- When the PKI administrator receives a certificate request from a preregistered SCEP client, the PKI administrator can confirm the integrity of the certificate request by viewing its fingerprints on the "Single Request" web page. (See Figure 69 on page 412 for a sample.)

  To ensure the integrity of the certificate request, the PKI administrator can contact the SCEP requestor to match the fingerprints in the received certificate request with the fingerprints in the original certificate request. (The certificate requestor can use the SCEP client software to view the fingerprints that are saved for the original request.)

Steps for enabling Simple Certificate Enrollment Protocol (SCEP)

Before you begin

The commands in the steps that follow include several variables that are described in Table 57 on page 337. Determine the values for these variables and record the information in the blank boxes:

Table 57. Information you need to enable Simple Certificate Enrollment Protocol (SCEP)

Information needed	Where to find this information	Record your value here
ca_label - The label of your CA certificate in RACF.	See Table 11 on page 41.	

Table 57. Information you need to enable Simple Certificate Enrollment Protocol (SCEP) (continued)

Information needed	**Where to find this information**	**Record your value here**
ra_label - The label of your RA certificate in RACF.	See Table 11 on page 41.	
ca_ring - The PKI Services SAF key ring.	See Table 19 on page 55.	
ca_expires - The date the PKI Services CA certificate expires.	See Table 19 on page 55.	
daemon - The user ID for the PKI daemon.	See Table 19 on page 55.	
ra_backup_dsn - The name of the encrypted data set containing the backup copy of your new RA certificate and private key.	See Table 19 on page 55.	
ra_dn - The RA's distinguished name.	See Table 19 on page 55.	

Procedure

Perform the following steps to enable PKI Services to process Simple Certificate Enrollment Protocol (SCEP) requests:

1. (Optional) Create your PKI Services RA certificate by following these steps, if you have not done so already. (This is optionally done by IKYSETUP.) If you already created an RA certificate, skip to Step "2" on page 338.

   a. To create an RA certificate, execute the following RACF command from the TSO command line:

   ```
   RACDCERT ID(daemon) GENCERT SUBJECTSDN(ra_dn) KEYUSAGE(HANDSHAKE)
     SIGNWITH(CERTAUTH LABEL('ca_label')) NOTAFTER(DATE(ca_expires))
     WITHLABEL('ra_label')
   ```

   b. Back up the new PKI Services RA certificate and private key to a password-encrypted data set (*ra_backup_dsn*). Remember to record and store your encryption password in case you ever need to recover the certificate or private key.

   ```
   RACDCERT ID(daemon) EXPORT(LABEL('ra_label')) DSN(ra_backup_dsn)
     FORMAT(PKCS12DER) PASSWORD('encryption-pw')
   ```

   c. Add the new RA certificate to the PKI Services key ring.

   ```
   RACDCERT ID(daemon) CONNECT(LABEL('ra_label') RING(ca_ring))
   ```

2. Edit the PKI Services configuration file (`/etc/pkiserv.conf`) and set the `RALabel` directive in the **SAF** section to specify the label (*ra_label*) of your PKI Services RA certificate. (The default in IKYSETUP is `Local PKI RA`. For details, see "(Optional) Steps for updating the configuration file" on page 74.)

   ```
   [SAF]
   KeyRing=PKISRVD/CAring
   # The label of the PKI Services RA certificate
   RALabel=Local PKI RA
   ```

3. Edit the PKI Services configuration file (`/etc/pkiserv.conf`) to change the EnableSCEP directive in the **CertPolicy** section setting from **F**(False) to **T**(True).

```
[CertPolicy]
# Enable the Simple Certificate Enrollment Protocol, (T)rue or (F)alse
EnableSCEP=T
```

4. Edit the PKI Services template file (`/etc/pkiserv.tmpl` or `pkitmpl.xml`) and customize the **<PREREGISTER>** section of the `5-Year SCEP Certificate - Preregistration` template as you want or create a new preregistration template. (Refer to the list in "Variables used in the <PREREGISTER> section" on page 335 for valid variables and values.

   (defaults)

```
AuthenticatedClient=AutoApprove
SemiauthenticatedClient=AdminApprove
UnauthenticatedClient=Reject
SubsequentRequest=AutoApprove
RenewalRequest=AutoApprove
```

5. Edit the **<CONTENT>** section of your preregistration template to allow the PKI administrator to specify subject distinguished name and alternate name fields that the SCEP client must provide to authenticate. Specify only subject distinguished name and alternate name fields here. All other fields are ignored. (For about customizing the end-user web pages, see Chapter 11, "Customizing the end-user web application if you use REXX CGI execs," on page 135.)

   (defaults)

```
%%SerialNumber (Optional)%%
%%UnstructAddr (Optional)%%
```

6. Edit the **<CONSTANT>** section of your preregistration template to supply any other value you want, such as MAIL or ORG, that must be included for every SCEP preregistration request. Any subject distinguished name and alternate name fields you specify here must match the information (in the subsequent certificate request) sent by the SCEP client to authenticate the certificate request.

```
%%Org=The Firm%%
```

7. Edit the HTTP Server environment variables file, `vhost80.conf` file, and update the LIBPATH variable to include `/usr/lpp/pkiserv/lib`.
8. Stop and restart PKI Services.

When you are done, you have enabled your CA domain to accept SCEP preregistration requests and process certificate requests from preregistered SCEP clients. The URL used by the SCEP clients is http://*webserver-domain-name*/*CA-domain-name*/public-cgi/pkiclient.exe.

Chapter 16. Using Enrollment over Secure Transport (EST)

Enrollment over Secure Transport (EST) is the successor to Simple Certificate Enrollment Protocol (SCEP), initially sponsored by Cisco. SCEP has not be standardized. EST is standardized by RFC7030, which profiles certificate enrollment for clients using Certificate Management over CMS (CMC) messages over a secure transport (RFC5272 - updated by RFC6402). This protocol aims to provision certificates in a more robust manner than the traditional SCEP. It also supports ECC certificates. Cisco IOS Software and Cisco IOS XE support EST.

PKI Services supports the following EST functions:

1. **cacerts**: requests the EST CA certificates (the whole chain of the issuers' certificates) using Simple PKI Request.
2. **simpleenroll**: requests a certificate with supplied public key using Simple PKI Request.
3. **simplereenroll** : requests a renew or rekey certificate with supplied public key using Simple PKI Request.

Set up PKI Services as an EST CA

In order to be an EST CA, the CA certificate must meet one of the following requirements (see RFC7030 section 3.6.1 and RFC6125 section 6.4).

- – contains the `id-kp-cmcRA` extended key usage extension, or
  – contains Domain or IP in the Subject Alternate Name extension with a value that will match the host name or the IP address in URIs provided by incoming EST requests, or
  – contains a Common Name in the Subject Distinguished Name with a value that will match the host name in URIs provided by incoming EST requests, if there is no Domain or IP is present in the Subject Alternate Name extension .

You may use the `IKYSETUP` script to set up the EST CA certificate.

Steps to enable EST processing on the PKI EST CA instance

1. Create the EST CA certificate, or use an existing CA if it fulfills the EST CA requirements.
2. Edit the PKI Services configuration file, `preserver.conf` to specify the values for the following keywords in the Cert Policy section:
   a. EnableEST=T
   b. ESTCAFile=<full pathname containing the EST CA file in DER format>
   c. ESTTemplate=<EST template nickname> as used in `pkiserv.tmpl` file or pkitmpl.xml
3. If you are using the CGI scripts, update the `pkiserv.tmpl` file and the HTTP configuration files:
   a. Edit the **<CONTENT>** section of your EST preregistration template to specify the subject distinguished name and the alternate name fields that the EST client must provide to authenticate itself. See "Preregistering EST client" on page 342.
   b. Edit the HTTP Server `vhost443.conf` file and `vhost1443.conf` file to update the `ScriptAliasMatch` entries that would map to the URI used by the EST client.
   c. Start HTTP Server.
4. If you are using the JSPs, update the pkitmpl.xml file:

a. Edit the initvalue in the tags with formtype InstallationSpecified to specify the subject distinguished name and the alternate name fields that the EST client must provide to authenticate itself. See "Preregistering EST client" on page 342.

b. Run the TemplateTool utility to convert the pkitmpl.xml file to the pkixgen.tmpl file.

c. Start the Liberty PKI server.

When you are done, you have enabled your CA domanin to accept EST requests from the EST clients.

Preregistering EST client

To request certificates using EST for the first time through the `simpleenroll` function, an EST client must be either preregistered or authenticated itself with a certificate to the EST capable CA. The PKI administrators can preregister EST clients in batches using the `pkiprereg` utility (See"Using the pkiprereg utility" on page 437) or administrators can preregister individual EST clients (one client at a time) using the end-user web pages.

When the PKI administrator preregisters an EST client using the end-user web page, they can use the EST (preregistration) certificate template called a `2-Year EST Certificate Preregistration`. (See "Steps for preregistering an SCEP or EST client" on page 403 for more information.) The PKI administrator fills out the request form by specifying the device or client name of the EST client, a passphrase, and additional (optional) subject name and alternate name information. The information used will match the client's EST request.

If the CGI path is being used, customize the **<CONSTANT>** section of the EST (preregistration) certificate template in the `pkiserv.tmpl` file to supply the additional optional information.

If the JSP path is being used, customize the `initvalue` in the tags with the `InstallationSpecified` formtype in the `pkitmpl.xml` file to supply the additional optional information. These fields will be created as entries under the **<CONSTANT>** section in the converted `pkixgen.tmpl` file.

When a PKI administrator submits the form for an EST (preregistration) certificate request, PKI Services creates a preregistration record, not an actual certificate request in the `ObjectStore` (request database). The client name is translated to lowercase characters, truncated to 32 characters if longer, and saved as the requestor name to support searching of the `ObjectStore`. (Each preregistration record must have a client name that is unique in the first 32 characters, regardless of uppercase or lowercase).

The preregistration record contains the template nickname, passphrase, and additional (optional) subject name and alternate name values. Any other information (unrelated to the subject name or alternate name) specified on the request form is ignored.

The fields in the **<CONSTANT>** section in the `pkiserv.tmpl` file or in the converted `pkixgen.tmpl` file that supplying additional (optional) values for the following variables are not used in the preregistration record. However, those values are processed when the preregistered client then requests a certificate.

- AuthInfoAcc
- CertPolicies
- Critical
- ExtKeyUsage
- KeyUsage
- NotAfter
- NotBefore

Processing an EST request

EST transfers CMC messages through a TLS-secured HTTP session. The messages used are Simple PKI Request and Simple PKI Response. The Simple PKI Request is a PKCS#10 certificate request message. The Simple PKI Response is in the format of CMS `ContentInfo`.

The EST message is processed by a CGI program called `pkiest`. The communication between the EST client and the CGI is over HTTPS only, the `cacerts` function uses the HTTP GET method, and all other functions require the HTTP POST method.

If the CA is running with a domain name, the domain name can be included in the URI for the CGI program. The domain name specified in the URI is used to locate the directory path of the `pkiserv.envars` file through the _PKISERV_CONFIG_PATH_<ca_domain> environment variable. If no domain value exists in the URI, then the _PKISERV_CONFIG_PATH environment variable is used.

From the `pkiserv.envars` file, the CA domain value is determined by the `_PKISERV_CA_DOMAIN` environment variable. If the domain name specified in the URI is different than that specified by `_PKISERV_CA_DOMAIN`, the latter value will be used.

The following examples of the URIs can be used by an EST client:

1. When the EST client needs to get the EST CA certificates, it uses the URI with the server authentication port.

   Note: This is usually the first step the EST client needs if it has not already obtained the CA certificate(s) from other means.

   `https://www.YourCompany.com:<server authentication port>/.well-known/est/cacerts` or `https://www.YourComapny.com:<server authentication port>/.well-known/est/<ca domain>/certs`
2. When the EST client needs to request a certificate from the EST CA, it uses the URI with the server authentication port.

   `https://www.YourCompany.com:<server authentication port>/.well-known/est/simpleenroll` or `https://www.YourComapny.com:<server authentication port>/.well-known/est/<ca domain>/simpleenroll`
3. When the EST client needs to renew a certificate from the EST CA, it uses the URI with the client authentication port.

   `https://www.YourCompany.com:<server authentication port>/.well-known/est/simplerenroll` or `https://www.YourComapny.com:<server authentication port>/.well-known/est/<ca domain>/simplereenroll`

Once the EST client is preregisterd by the PKI administrator, the EST client can request a certificate. When the preregistered EST client requests a certificate (an EST simpleenroll request is sent), PKI Services searches for a preregistration record that matches the client name. If one is found, PKI Services compares the values in the request to challenge the password and any subject name or alternate name information specified by the PKI administrator or supplied in the **<CONSTANT>** template section.

If a matching preregistration record is found, a regular PKI request will be created and the preregistration record will be deleted. Otherwise, the preregistration record will remain untouched.

The EST client can also renew the certificate that was obtained previously through the simpleenroll process. The function for renew is simplereenroll and no preregistration step is needed. The original certificate is all that is needed to authenticate in the simplereenroll process through the TLS client-authentication protocol.

Tracing the PKI EST program

The administrator can enable tracing of the PKI Services EST CGI program using the environment variables _PKISERV_EST_TRACE to set the trace option and _PKISERV_EST_TRACE_FILE to specify the name of the trace file in the `vhost443.conf` and `vhost1443.conf` files.

Table 58. HTTP Server environement variables used to control tracing.

HTTP Server environment variable	Description
_PKISERV_EST_TRACE	Specifies a bit mask enabling EST trace options. The bit mask can be specified as a decimal (*nnn*), octal (*0nnn*), or a hexademical (*0xhh*) value. These trace options are available: **0x00** No trace. **0x01** EST error messages. **0x02** EST informational messages. **0x04** R_PKIServ callable service parameter list traces on entry and exit. **0x08** Elapse time messages of events within the EST program. **0x10** EST program function entry and exit trace messages. **0x20** DER buffer display messages. **0x40** Displays environment variables that are set at EST program startup. **Example:** `SetEnv _PKISERV_EST_TRACE 0x7f`
_PKISERV_EST_TRACE_FILE	Specifies the name of the trace file. Defaults to */tmp/pkiest.%.trc*. The trace file is not used if the _PKISERV_EST_TRACE environment variable is not defined or is set to 0. The current process identifier is included as part of the trace file name when the name contains a percent sign (%). For example, if _PKISERV_EST_TRACE_FILE is set to */tmp/pkiest.%trc* and the current process identifier is i123, the trace file name is */tmp/pkiest.123.trc*. **Guideline:** Because multiple copies of the EST CGI program can run concurrently for multiple EST clients, the value of _PKISERV_EST_TRACE_FILE should include the percent sign (%) to prevent multiple copies of the EST CGI program from writing to the same file. **Example:** `SetEnv _PKISERV_EST_TRACE_FILE /tmp/pkiest.%.trc`

Messages and codes returned from the EST functions

Failure conditions detected by the pkiest CGI program are returned to the EST client application as an HTTP status code and an internal pkiest error code with a descriptive message explaining the cause of the failure condition. The error code and descriptive message are also recorded to the IBM HTTP Server error_log file. Details of the execution path that led to encountering the failure condition are recorded to the trace file if tracing has been activated through the appropriate environment variables. See "Tracing the PKI EST program" on page 343 for more information.

Table 59. HTTP status code, CGI error code and the description

CGI error code	HTTP status	Description
(06)	500	Request queue serialization timeout occurred.
(08)	401	Request denied, not authorized.
(12)	500	An internal error has occurred during RACF processing.
(20)	500	Function code specified is not defined.
(28)	503	Certificate generation provider not available for specified CA domain: *domain*
(32)	500	Incorrect value specified for CA domain.
(52)	500	*CertPlist* has an incorrect value.
(56)	500	Required field *fieldname* is missing from the request.
(60)	500	Certificate generation provider indicated the following error: *error*.
(64)	400	Certificate could not be verified by CA domain: *domain*.
(72)	500	Certificate could not be reenrolled because of state change in the CA domain: *domain*.
(76)	500	Conflicting field names in CertPlist.
(80)	501	Enrollment over Secure Transport EST) disabled: *error*.
(84)	401	No preregistration record found for the EST request or unable to authenticate it: *error*.
(99)	500	*R_PKIServ-function* Failed, safrc=*safrc*, racfrc=*racfrc*, racfrsn=*racfrsn*.
(464453637)	500	VSAM contention caused the request to fail. Retry the request.
(464453634)	500	VSAM contention caused the request to fail. Retry the request.
(5001)	500	Failed to create EST response message, status=*error code*.
(5002)	500	Base64 encode of output message failed.
(5003)	403	Error occurred, HTTP access is forbidden.
(5004)	405	Error occurred, HTTP method was *HTTP method name* instead of *method name*.
(5005)	500	*envar name* envar value length is greater than the maximum length of *maximum length*.
(5006)	500	Erro occurred attempting to read the HTTP input message.
(5007)	500	Storage allocation failed *element:size*.
(5008)	500	Exported PKCS7 package is empty.
(5009)	400	Premature end of data.
(5010)	500	FIPS state setting failed.
(5011)	400	ASN.1 data discrepancy: *description*.
(5012)	500	gsk function *function* failed, error code = *error-code*.

Table 59. HTTP status code, CGI error code and the description (continued)

CGI error code	HTTP status	Description
(5013)	401	CA cert is not an EST CA certificate.
(5014)	500	Exported ESTCA does not match local ESTCA.
(5015)	400	No CA domain found.
(5016)	501	EST is not enabled.
(5017)	400	Method name not found from URI.
(5018)	401	Challenge Passphrase not provided.
(5019)	400	Unsupported data size.
(5020)	500	Error processing PKI Services with configuration file (*configuration-file-name*).
(5021)	501	CA domain *domain-name* does not have EST support enabled.
(5022)	500	pkcs7 buffer not supplied on input.
(5023)	500	Gencert succeeded, but not Transaction ID returned.
(5024)	400	Method name *method-name* is not a valid method name or is not supported.
(5025)	400	Request length is 0.
(5026)	500	Response length is 0.
(5027)	500	Failure validating CA certificate
(5028)	500	Error getting local ESTCA file.
(5029)	500	Could not verify ESTCA file.
(5030)	500	Invalid parameter.
(5031)	400	URI unreadable.
(5032)	400	.well-known/est/ not detected in path name.
(5033)	400	.well-known/est/ at end of path name.
(5034)	500	ESTCA file was not found at *file-location* or could not be read.
(5035)	500	Error returning certificate.
(5036)	400	URI contains extraneous text.
(5037)	500	Unable to extract *fieldname* field from summary list.
(5038)	400	Passphrases in the request do not match: challengePASSWORD, estIdenityLinking.
(5039)	500	Error extracting attribute.
(5040)	400	Error occurred, content type was *content-type* instead of *content-type*.
(5041)	500	Failure to convert *address* to numeric address.
(5042)	400	PKCS10 CSR validation failure, error *error*.
(5043)	401	Failure validating client certificate: *System SSL function* error *errorcode-errordescription*.
(5045)	400	EST request does not identify subject in client certificate.

Table 59. HTTP status code, CGI error code and the description (continued)

CGI error code	HTTP status	Description
(5046)	500	*Environment* / *template* file could not be processed - error *error*.
(5047)	400	*Environment* / *template* file could not be processed - error *error*.
(5048)	500	Template *template* incorrectly formatted.

Chapter 17. Customizing with installation exit routines

PKI Services supports the use of installation exit routines in the following ways:

- The PKI Services daemon can call an installation-provided exit routine for automatic renewal processing.
- If you implement the PKI Services web application using REXX CGI execs, the PKI Services web application CGIs can call an installation-provided exit routine for end-user functions except VERIFY.
- If you implement the PKI Services web application using JavaServer pages (JSPs), exit methods are called before and after end-user functions except VERIFY.

PKI Services provides a sample exit routine, `pkiexit.c`, written in the C language. It is intended to demonstrate the power of the exit routine for the daemon and REXX CGI execs, and to provide a guide for you to write your own exit routine. The main routine of the program determines which subroutine to call, based on the `R_PKIServ` function being called and whether this is a pre- or post-processing call.

No sample is provided for the exit methods used with JSPs.

You can implement the exit routines for the PKI Services daemon and the PKI Services web application CGIs in the same program (as shown in the sample, `pkiexit.c`) or in separate programs.

PKI Services provides the following files for the daemon and CGI exit routine. Both files are, by default, in: `/usr/lpp/pkiserv/samples/`.

Table 60. Files for the `pkiexit.c` exit routine

File name	Description
`pkiexit.c`	Code sample for the exit routine (in the C programming language). You probably need to update the exit routine code before using it.
`Makefile.pkiexit`	Makefile for `pkiexit.c`.

Exit routine processing for automatic certificate renewal

The PKI Services daemon supports an installation-provided exit routine for automatic renewal processing. An exit routine can be written to provide additional automatic renewal criteria, and to capture the renewed certificate for further processing. If you choose to implement this exit routine, it must be a UNIX executable program residing in a file system, with appropriate permission assigned. The PKI Services daemon identifies the exit routine as the program specified by the value of the _PKISERV_EXIT environment variable in the `pkiserv.envars` file. The value that is specified is limited to a maximum of 256 characters. The exit routine is invoked by the PKI Services daemon using standard UNIX parameters (that is, `argc` and `argv`[]). The exit routine communicates its results back to the PKI Services daemon by way of a return code. The exit routine is called for preprocessing and post-processing before and after automatic certificate renewal processing. Unlike the PKI Services CGI exit routines, messages that are written to either STDOUT or STDERR do not appear in either the web server or PKI Services daemon logs. If you want to write messages in the exit program, you need to open a file and write messages to that file. The sample exit routine that is provided in `/usr/lpp/pkiserv/samples/pkiexit.c` illustrates writing messages to a file in both the preprocessing and post-processing exit functions.

Note: This exit routine can be implemented in the same program as the exit routines for the PKI Services CGIs (as illustrated in the sample `pkiexit.c` exit program) or can be implemented as a separate program.

The `ExitTimeout` keyword in the General section of the `pkiserv.conf` file specifies the maximum time PKI Services waits for the exit routine to return. If `ExitTimeout` is not specified, PKI Services waits at most 30 seconds for the exit routine to return. If `ExitTimeout` is specified with a value greater than 1 hour, PKI Services waits 1 hour at the most for the exit routine to return.

Steps for updating the exit routine code sample

To update the exit routine code sample, `pkiexit.c`, perform the following steps:

1. Copy the sample exit routine and makefile to the current directory by entering the following commands:

   ```
   cp /usr/lpp/pkiserv/samples/pkiexit.c pkiexit.c
   cp /usr/lpp/pkiserv/samples/Makefile.pkiexit Makefile
   ```

   ____
2. Compile and link to produce the executable program, `pkiexit`, by entering the following command:

   ```
   make
   ```

   ____
3. Move the executable program to its execution directory and set the permissions by entering the following commands:

   ```
   mv pkiexit /full-directory-name
   chmod 755 /full-directory-name/pkiexit
   ```

   ____
4. Edit the web server environment variables to include `_PKISERV_EXIT`, and set its value to the full path name of the exit program. You may set the environment variable in each of the virtual host configuration files or in the main `httpd.conf` file with a `SetEnv` directive. For example, if the exit program is called `pkiexit` and is in the `/usr/local/bin` directory, the `SetEnv` directive would be entered as follows:

   ```
   SetEnv    _PKISERV_EXIT   /usr/local/bin/pkiexit
   ```

Using the exit routine for pre- and post-processing

This exit routine is called for preprocessing and postprocessing by the PKI Services daemon before and after it renews a certificate respectively. Automatic certificate renewal processing is performed by the daily maintenance task. By default this task runs when the PKI Services daemon is started and every day at midnight, but you can customize the days on which it runs and the time at which it runs. (For information about customizing when the maintenance task runs, see "Optionally updating the pkiserv.conf configuration file" on page 72.)

Table 61. Values of arguments for pre- and post-processing

Time of processing	Argument 1	Argument 2
Preprocessing	0	The function number in EBCDIC: **500** automatic renewal processing
Post-processing	1	The function number in EBCDIC: **500** automatic renewal processing

Automatic renewal - preprocessing

Purpose

Provide additional criteria for automatic renewal of certificates.

Arguments

***argument* 3**
The Base64-encoded original certificate.

Return codes

Return code	Meaning
0	Continue with the request for automatic renewal processing.
4	Disallow automatic renewal processing for the current request.
8	Disallow automatic renewal processing from now on.

Note: Any value other than 0, 4, and 8 is treated as 4.

STDOUT

Non-applicable.

Note: The automatic renewal exit routine cannot write to STDOUT or STDERR like other exit routines. The output must be written to a file.

Automatic renewal - post-processing

Purpose

Capture the renewed certificate for further processing.

Arguments

***argument* 3**
The Base64-encoded renewed certificate.

Return codes

Return code	Meaning
0	Normal return.

Note: The return code is not checked from the post exit processing.

STDOUT

Non-applicable.

Note: The automatic renewal exit routine cannot write to STDOUT or STDERR like other existing exit routines. The output must be written to a file.

Scenario for using the exit routine

This scenario disables the automatic renewal of certificates for contractors, postpones the renewal if the expiration date is more than 30 days away, and logs the subject name and serial number of certificates that are automatically renewed.

For sample code illustrating this scenario, see the sample exit routine `pkiexit.c` shipped with PKI Services.

The preprocessing exit routine for the automatic renewal function (subroutine `preProcessAutoRenewExit`) disables the automatic renewal of certificates for contractors and postpones the renewal if the expiration date is more than 30 days away. Here are the steps:

- Get the current time and format it for output.
- Call subroutines to decode the Base64 certificate data and decode the certificate.
- Get a printable version of the subject name from the certificate.
- Get a printable version of the serial number from the certificate.
- Check the subject name for an organizationalUnitName of "Contractors". If found, log a message indicating that the renewal was disabled, and return with a return code of 8 to disable the automatic renewal.
- Call a subroutine to determine how many days there are until the certificate expires.
- If there are more than 30 days before the certificate expires, log a message indicating that the renewal was postponed, and return with a return code of 4 to postpone the automatic renewal.
- If there are 30 or fewer days before the certificate expires, log a message indicating that renewal of the certificate was allowed, and return with a return code of 0 to continue with the automatic renewal.

The postprocessing exit routine for the automatic renewal function (`postProcessAutoRenewExit`) logs the subject name and serial number of certificates that are automatically renewed. Here are the steps:

- Get the current time and format it for output.
- Call subroutines to decode the Base64 certificate data and decode the certificate.
- Get a printable version of the subject name from the certificate.
- Get a printable version of the serial number in hexadecimal from the certificate.
- Log a message indicating that the certificate was renewed, containing the subject name, serial number, and time.

Exit routine processing for the PKI Services CGIs

For the end-user functions except VERIFY, the PKI Services web application CGIs support calling an installation-provided exit routine. The exit routine can perform tasks such as:

- Provide additional authorization checking
- Validate and change parameters
- Capture certificates for further processing
- Recover a passphrase that is used in a certificate request

If the exit routine exists, it must be a UNIX executable program residing in the file system, and it must have appropriate permission assigned. To specify the exit routine, the UNIX programmer sets the _PKISERV_EXIT environment variable in the web server's environment variables file. The environmental variable may also be added to the web server's configuration file (`httpd.conf`) by using the SetEnv HTTP Directive. On input, it receives standard UNIX parameters (that is, *argc* and *argv* `[]`). It communicates back to PKISERV through the return code and by writing to STDOUT.

Steps for updating the exit routine code sample

To update the exit routine code sample, `pkiexit.c`, perform the following steps:

1. Copy the sample exit routine and makefile to the current directory by entering the following commands:

```
cp /usr/lpp/pkiserv/samples/pkiexit.c pkiexit.c
cp /usr/lpp/pkiserv/samples/Makefile.pkiexit Makefile
```

2. Compile and link to produce the executable program, `pkiexit`, by entering the following command:

```
make
```

3. Move the executable program to its execution directory and set the permissions by entering the following commands:

```
mv pkiexit /full-directory-name
chmod 755 /full-directory-name/pkiexit
```

4. Edit the web server environment variables to include _PKISERV_EXIT, and set its value to the full path name of the exit program. You may set the environment variable in each of the virtual host configuration files or in the main `httpd.conf` file with a `SetEnv` directive. For example, if the exit program is called `pkiexit` and is in the `/usr/local/bin` directory, the `SetEnv` directive would be entered as follows:

```
SetEnv    _PKISERV_EXIT   /usr/local/bin/pkiexit
```

Using the exit routine for pre- and post-processing

The exit routine is called:

- For preprocessing before calling the R_PKIServ (IRRSPX00) SAF callable service.
- For post-processing after returning from the callable service.

The following table summarizes the values of the first two arguments for pre- and post-processing. (Additional arguments vary, depending on the function to perform.)

Table 62. Values of arguments for pre- and post-processing

Time of processing	Argument 1	Argument 2
Preprocessing	0	The function number from the R_PKIServ SAF callable service in EBCDIC: **1** GENCERT **2** EXPORT **9** REQCERT **11** REVOKE **12** GENRENEW **13** REQRENEW **17** QRECOVER

Table 62. Values of arguments for pre- and post-processing (continued)

Time of processing	Argument 1	Argument 2
Post-processing	1	The function number from the R_PKIServ SAF callable service in EBCDIC: **1** GENCERT **2** EXPORT **9** REQCERT **11** REVOKE **12** GENRENEW **13** REQRENEW **17** QRECOVER

Note: The parameters that are input to the CGIs and the values that are resolved by the CGIs (`argument` **3**…`argument` *n* for all functions) vary based on how you have customized the templates.

Return codes

The topics that follow contain tables of expected return codes. If calling the exit routine produces an unexpected return code, that is, one that is not listed, PKI Services treats it as a failure. Processing for the request stops and an error message is issued.

The return code is a one-byte value.

GENCERT and GENRENEW - preprocessing

Purpose

Provide additional authorization checking, parameter validation and modification, and a mapping of the passphrase to a set of security answers to use during QRECOVER processing, if the passphrase is forgotten.

Arguments

argument* 3...*argument n
The parameters as input to the CGI plus values resolved by the CGI in *name=value* form, for example, `"CommonName=Sam Smith"`.

Return codes

Return code	Meaning
0	Continue with the request with possible modifications.
4	Continue with the request with possible modifications, but change it to require administrator approval.
8 - 49	Deny the request and return to the caller immediately.

STDOUT

Zero or more additional *CertPlist* parameters to add to the request in *name=value* form, one per line. For those fields defined as non-repeating (according to the documentation for the IRRSPX00 callable service, for example, *CommonName*), specifying the parameters here in effect replaces the CGI input values.

GENCERT and GENRENEW - post-processing

Purpose

Capture the TransactionId or failing return codes for further processing.

Arguments

***argument* 3...*argument n*–3**
: The final set of parameters as determined by the preprocessing exit in *name=value* form.

***argument n*–2**
: The RACF return code from the callable service.

***argument n*–1**
: The RACF reason code from the callable service.

argument n
: The *TransactionId*. This is a string of undetermined value if the request was unsuccessful.

Return codes

Return code	Meaning
0	Normal

STDOUT

Optional replacement *TransactionId*.

REQCERT and REQRENEW - preprocessing

Purpose

Provide additional authorization checking, parameter validation and modification, and a mapping of the passphrase to a set of security answers to use during QRECOVER processing, if the passphrase is forgotten.

Arguments

argument* 3...*argument n
: The parameters as input to the CGI plus values resolved by the CGI in *name=value* form, for example, "`CommonName=Sam Smith`".

Return codes

Return code	Meaning
0	Continue with the request with possible modifications.
4	Continue with the request with possible modifications, but change it to not require administrator approval.
8 - 49	Deny the request and return to the caller immediately.

STDOUT

Zero or more additional *CertPlist* parameters to add to the request in *name=value* form, one per line. For those fields defined as non-repeating (according to the documentation for the IRRSPX00 callable service, for example, *CommonName*), specifying the parameters here in effect replaces the CGI input values.

REQCERT and REQRENEW - post-processing

Purpose

Capture the *TransactionId* or failing return codes for further processing.

Arguments

***argument* 3...*argument n*–3**
: The final set of parameters as determined by the preprocessing exit routine in *name=value* form.

***argument n*–2**
: The RACF return code from the callable service.

***argument n*–1**
: The RACF reason code from the callable service.

argument n
: The *TransactionId*. This is a string of undetermined value if the request was unsuccessful.

Return codes

Return code	Meaning
0	Normal

STDOUT

Optional replacement *TransactionId*.

EXPORT - preprocessing

Purpose

Provide additional authorization checking and parameter validation and modification.

Arguments

argument* 3...*argument n
: The parameters as input to the CGI in *name=value* form, for example, "`TransactionId=12345`".

Return codes

Return code	Meaning
0	Continue with the export.
8 - 49	Deny the request and return to the caller immediately.

STDOUT

Optional replacement *TransactionId* and *ChallengePassPhrase* parameters in *name=value* form, one per line. If these values are provided, they replace the user-provided values on the call to the SAF callable service. If TransactionId is specified without *ChallengePassPhrase*, the user-provided

ChallengePassPhrase is used. If *ChallengePassPhrase* is specified without *TransactionId*, the user-provided *TransactionId* is used.

EXPORT - post-processing

Purpose

Capture the certificate or failing return codes for further processing.

Arguments

***argument* 3...*argument n*–3**
The parameters as input to the CGI in *name*=*value* form, followed by any modified value provided by the preprocessing exit routine, also in *name*=*value* form.

***argument n*–2**
The RACF return code from the callable service.

***argument n*–1**
The RACF reason code from the callable service.

argument n
The base64-encoded certificate with header and footer. This is a string of undetermined value if the request was unsuccessful.

Return codes

Return code	Meaning
0	Normal

STDOUT

Non-applicable.

REVOKE - preprocessing

Purpose

Provide additional authorization checking and parameter validation.

Arguments

argument* 3...*argument n
The parameters as input to the CGI in *name*=*value* form, for example, "`reason=1`".

Return codes

Return code	Meaning
0	Continue with the request.
8 - 49	Deny the request and return to the caller immediately.

STDOUT

Non-applicable.

REVOKE - post-processing

Purpose

Capture the certificate or failing return codes or both for further processing.

Arguments

***argument* 3...*argument n*–2**
The parameters as input to the CGI in *name*=*value* form, for example, "`reason=1`".

***argument n*–1**
The RACF return code from the callable service.

argument n
The RACF reason code from the callable service.

Return codes

Return code	Meaning
0	Normal

STDOUT

Non-applicable.

QRECOVER - preprocessing

Purpose

Provide a mechanism to retrieve a passphrase needed to recover a certificate, in case it was forgotten. The exit routine is called when a user has entered answers to the security questions instead of a passphrase.

Arguments

argument* 3...*argument n
The parameters as input to the CGI in *name*=*value* form, for example, "`Security1=Brazil`".

Return codes

Return code	Meaning
0	Continue with the query.
4	Cannot determine the passphrase.
8 - 49	Deny the request and return to the caller immediately.

STDOUT

Optional replacement of `Requestor` and add the `PassPhrase` parameters in `name=value` form, one per line.

QRECOVER - post-processing

Purpose

Capture the list of the recovery certificates for further processing.

Arguments

***argument* 3...*argument n*–3**
The input and output set of parameters of the preprocessing exit routine in the *name*=*value* form.

***argument n*–2**
The RACF return code from the callable service.

***argument n*–1**
The RACF reason code from the callable service.

argument n
The list of the recovery certificates.

Return codes

Return code	Meaning
0	Normal

STDOUT

Non-applicable.

Scenarios for using the exit routine

The sample exit routine supplied with PKI Services, `pkiexit.c`, illustrates the following scenarios. The main routine of the program determines which subroutine to call, based on the `R_PKIServ` function being called and whether this is a pre- or post-processing call. Individual subroutines in the program handle the scenarios.

Scenario 1: Allow only selected users to request PKI browser certificates for authenticating to z/OS

This scenario is for allowing only selected local z/OS users to request PKI browser certificates for authenticating to z/OS. Additionally, this scenario is for providing a customized TITLE value for the subject's distinguished name based on the user's role in the organization. Permission and the user's role in the organization is indicated by access to the BPX.SERVER resource in the FACILITY class and by the user's level of access to FACILITY class resources called PROJ.MEMBER and PROJ.PARTNER. The access values are as follows:

NONE
No access for either resource. The user is not permitted to request this type of certificate. The certificate request is denied.

READ to PROJ.MEMBER
The user is a team member and is permitted to request the certificate. The TITLE value is set to `Team Member`. Certificate requests for team members are automatically approved. (No administrator approval is required.)

UPDATE to PROJ.MEMBER
The user is the team's leader and is permitted to request the certificate. The TITLE value is set to `Team Leader`. A certificate request by the team leader is automatically approved. (No administrator approval is required.)

READ to PROJ.PARTNER

The user is considered to be a general partner of the team, not an active team member. The user is allowed to request certificates, but the requests require administrator approval before being issued. The TITLE value is set to `Team Partner`.

UPDATE to PROJ.PARTNER

The user is considered to be a trusted partner of the team, not an active team member. The user is allowed to request certificates, and unlike requests of the general partner, the certificate request are automatically approved. The TITLE value is set to `Team Trusted Partner`.

The preprocessing exit routine call for the GENCERT and REQCERT functions (subroutine `preProcessGenReqCertExit`) handles the logic described in the preceding. Here are the steps:

- The request values are passed into the exit routine through *argv* in *field-name=field-value* pairs, and the subroutine looks for the `Template=` and `UserId=` in the input parameters.
- When the exit routine code finds a `Template=` value containing `PKI Browser Certificate For Authenticating To z/OS`, the `__check_resource_auth_np()` system function examines the user ID. This determines the user's access to the preceding profiles.
  - If the user has no authority to either of these resources, return code 8 is set. This causes the request to be denied.
  - Otherwise the user's TITLE is set by writing the `TITLE=`*`title-value`* string to STDOUT.

  By default, administrator approval is not required for the PKI browser certificate for authenticating to z/OS.

  - When the user has only READ access to PROJ.PARTNER, the function must be changed to require administrator approval. This is done by setting return code 4.
  - For all other accesses the function does not need to be changed.

Scenario 2: Maintain a customized certificate repository (database) independent of PKI Services

This scenario is for maintaining a customized certificate repository (database) that is independent of PKI Services. After a successful submission of a certificate request, PKI Services returns the transaction ID. This is saved in a new customer-provided database entry. An alias for this database entry is then returned to the end user as the transaction ID. Later, when the user wants to pick up the certificate, the user-entered alias name is used to retrieve the actual PKI Services transaction ID. The retrieved certificate is saved in the database entry before being returned to the user.

Three different exit routine calls handle the preceding logic.

- Post-processing for the GENCERT or REQCERT functions (subroutine `postProcessGenReqCertExit`) returns a pretend alias entry name by suffixing the actual transaction ID with either SAF or PKI. This is where the database entry should be created. (Note that the exit routine performs no actual database calls because this would be too customer-specific.)
- Preprocessing for the EXPORT function (subroutine `preProcessExportExit`) reverts the transaction ID to its original value. This emulates retrieval from the database entry.
- Post-processing for the EXPORT function (subroutine `postProcessExportExit`) saves the returned certificate to a database entry. This is emulated by writing it to a file.

Scenario 3: Mandate a policy for certificate renewal only within 30 days of expiration

This scenario is for mandating a policy that allows users to renew their certificates only when certificates are within 30 days of expiring. When the condition is met, you can change the expiration date for the renew request so that the new certificate's validity period is extended by the number of days that are specified by the NotAfter parameter. In other words, the new certificate should expire *n* days from the current date, where *n = number of days remaining in the old certificate's validity period + number of days specified by NotAfter*.

The preprocessing exit routine call for GENRENEW and REQRENEW functions (subroutine `preProcessGenReqRenewExit`) handles the preceding logic. Here are the steps:

- The user's certificate is extracted from the environment variable `HTTPS_CLIENT_CERT`.
- The *NotAfter* value is extracted from the input parameters (*argv*), converted to a number, and saved in the variable *RequisitePro*.
- Subroutine `determineExpiration` is called to extract the expiration date from the user's certificate. This subroutine calls several subroutines to base64 decode the certificate, DER decode the binary certificate, and convert the expiration date to a seconds value.
- Upon return from `determineExpiration`, the variable *timeBeforeExp* is the number of seconds from now that the certificate expires. This is compared against the number of seconds in 30 days (86400 × 30) to see if it is greater than 30 days.
  - If it is greater than 30, the request is rejected by setting return code 8.
  - If it is not greater than 30, the new *NotAfter* value is computed as *timeBeforeExp*/86400 + *requestPeriod*.
- This new *NotAfter* value is set by writing it to STDOUT.

Scenario 4: Allow users to recover a PKI generated key certificate when the passphrase is lost

To recover a certificate for which PKI Services generated the keys, the user must provide the passphrase that was provided when the certificate was requested. This scenario illustrates how PKI Services can recover lost passphrases for PKI generated key certificates. To be able to recover a lost passphrase, the user must provide answers to security questions in addition to the passphrase when the user initially requests the PKI generated key certificate. PKI Services saves the passphrase and the answers to the security questions in a passphrase mapping database. To recover the lost passphrase, the user provides the answers to the security questions through the PKI Services web page. PKI Services searches the passphrase mapping database, and if the security answers match those provided by the user when the certificate was requested, the passphrase is returned to the CGI. The recovered passphrase is then used to retrieve the PKI generated key certificate.

Two exit routine calls are required:

- When the user requests the PKI generated key certificate, the preprocessing exit routine for the GENCERT and REQCERT functions (subroutine `preProcessGenReqCertExit`) collects the requestor name, the passphrase, and the answers to the security questions from the exit routine's parameter list. The exit routine records the information as an entry in a passphrase mapping database.
- When the user attempts to recover the PKI generated key certificate, the preprocessing exit routine for the QRECOVER function (subroutine `preProcessQRecoverExit`) collects the requestor name and the answers to the security questions from the exit routine's parameter list. The exit routine then searches the passphrase mapping database for entries that match the requestor name and the security answers provided by the user. If a match is found, the passphrase recorded in that entry is returned to the CGI through STDOUT.

Exit routine processing for JavaServer pages (JSPs)

If you implement the PKI Services web application using JavaServer pages (JSPs), you can use methods in the class com.ibm.pki.web.exits.UserExit to customize the web application. These methods are called before and after each of the following `R_PKIServ` requests:

- GENCERT
- REQCERT
- EXPORT
- GENRENEW
- REQRENEW
- REVOKE
- QRECOVER

The methods are empty stub methods in which you can add code to audit or modify parameters being passed to PKI Services. This topic describes these methods and related classes.

The R_PKIServ callable service is described in *z/OS Security Server RACF Callable Services*.

Table 63. Package and class summary for JSP exit processing

Package	Class
com.ibm.pki.web.exits	• UserExit • ExportCert • QRecover • RevokeCert • UserExitException
com.ibm.pki.rpkiserv	• CertPlist • PkiCertificate • QrecoverResultsList • RpkiservException

Class UserExit

Package: com.ibm.pki.web.exits

```
public class UserExit
extends java.lang.Object
```

UserExit defines the following static constant integer values:

```
UserExit.SUCCESSFUL = 0;
UserExit.CHANGE_APPROVAL_STATUS = 4;
UserExit.QRECOVER_PASSPHRASE_NOT_FOUND = 4;
UserExit.DENY_REQUEST = 8;
```

Table 64. Methods in class UserExit

Method	Purpose
preGenReqCert	Called before GENCERT or REQCERT requests
postGenReqCert	Called after GENCERT or REQCERT requests
preGenReqRenew	Called before GENRENEW or REQRENEW requests
postGenReqRenew	Called after GENRENEW or REQRENEW requests
preExport	Called before EXPORT requests
postExport	Called after EXPORT requests
preRevoke	Called before REVOKE requests
postRevoke	Called after REVOKE requests
preQRecover	Called before QRECOVER requests
postQRecover	Called after QRECOVER requests

preGenReqCert method

```
public int preGenReqCert(java.lang.String domain,
                         com.ibm.pki.rpkiserv.CertPlist plist,
                              java.lang.String[] security)
                      throws UserExitException
```

Purpose

Called before GENCERT or REQCERT requests.

Parameters

domain
: domain name

plist
: CertPlist with input parameters for GENCERT or REQCERT processing whose values can be modified by this method

security
: An array of responses to security questions. These correspond to form fields with names `security1`, `security2`, and so forth, in ascending numerical order.

returns

Value
: **Meaning**

0 (UserExit.SUCCESSFUL)
: Continue with the request.

4 (UserExit.CHANGE_APPROVAL_STATUS)
: If the certificate request required administrator approval, change it to not require administrator approval (a GENCERT). If the certificate request did not require administrator approval (a GENCERT), change it to require administrator approval (a REQCERT).

8 (UserExit.DENY_REQUEST) or greater
: Deny the request.

throws

Exception	Result
UserExitException	Handled the same as a return value of 8, and exception text is displayed on the resulting web page

postGenReqCert method

```
public java.lang.String postGenReqCert(java.lang.String domain,
                                       com.ibm.pki.rpkiserv.CertPlist plist,
                                       int RACFrc,
                                       int RACFrsncode,
                                       java.lang.String transactionid)
```

Purpose

Called after GENCERT or REQCERT requests

Parameters

domain
: Domain name

plist
: CertPlist that was input to R_PKIServ processing, including modifications made by the preGenReqCert method

RACFrc
RACF return code

RACFrsncode
RACF reason code

transactionid
Transaction ID, null if GENCERT processing was unsuccessful

returns

transaction ID

preGenReqRenew method

```
public int preGenReqRenew(java.lang.String domain,
                          com.ibm.pki.rpkiserv.CertPlist plist,
                          java.lang.String serialnum)
                   throws UserExitException
```

Purpose

Called before GENRENEW or REQRENEW requests

Parameters

domain
Domain name

plist
CertPlist with input parameters for renew request, whose values can be modified by this method

Note: Most values for a certificate renewal are taken from the existing certificate and therefore are not in the certPlist. The following values can occur in the certPlist:

- CertPlist.CERTPLIST_NOTIFYEMAIL
- CertPlist.CERTPLIST_PASSPHRASE
- CertPlist.CERTPLIST_NOTAFTER
- CertPlist.CERTPLIST_CERTPOLICIES
- CertPlist.CERTPLIST_AUTHINFOACC
- CertPlist.CERTPLIST_CRITICAL

serialnum
Serial number of certificate being renewed

returns

Value
Meaning

0 (UserExit.SUCCESSFUL)
Continue with the request.

4 (UserExit.CHANGE_APPROVAL_STATUS)
If the renewal request required administrator approval, change it to not require administrator approval (a GENRENEW). If the certificate request did not require administrator approval (a GENRENEW), change it to require administrator approval (a REQRENEW).

8 (UserExit.DENY_REQUEST) or greater
Deny the request.

throws

Exception	Result
UserExitException	Handled the same as a return value of 8, and exception text is displayed on the resulting web page

postGenReqRenew method

```
public java.lang.String postGenReqRenew(java.lang.String domain,
                                        com.ibm.pki.rpkiserv.CertPlist plist,
                                        int RACFrc,
                                        int RACFrsncode,
                                        java.lang.String transactionid)
```

Purpose

Called after GENRENEW or REQRENEW requests

Parameters

domain
Domain name

plist
The CertPlist that was input to the `R_PKIserv` request, including any modifications that were made by the preGenReqRenew method

RACFrc
RACF return code

RACFrsncode
RACF reason code

transactionid
Transaction ID, null if REQCERT processing was unsuccessful

returns

transactionid

preExport method

```
public int preExport(java.lang.String domain,
                     com.ibm.pki.rpkiserv.ExportCert exportobject)
```

Purpose

Called before EXPORT requests

Parameters

domain
Domain name

exportobject
ExportCert object containing transaction ID and passphrase

returns

Value
Meaning

0 (UserExit.SUCCESSFUL)
Continue with the EXPORT.

8 (UserExit.DENY_REQUEST) or greater
Deny the request.

throws

Exception	Result
UserExitException	Handled the same as a return value of 8, and exception text is displayed on the resulting web page

postExport method

```
public void postExport(java.lang.String domain,
                       com.ibm.pki.rpkiserv.ExportCert exportobject,
                       int RACFrc,
                       int RACFrsncode)
```

Purpose

Called after EXPORT requests

Parameters

domain
Domain name

exportobject
ExportCert object containing transaction ID and passphrase. The ExportCert object contains a base64-encoded certificate with header and footer if the request was successful.

preRevoke method

```
public int preRevoke(java.lang.String domain,
                     RevokeCert revokeobject)
```

Purpose

Called before REVOKE requests

Parameters

domain
Domain name

revokeobject
RevokeCert object containing reason number and serial number

returns

Value
Meaning

0 (UserExit.SUCCESSFUL)
Continue with the REVOKE request.

8 (UserExit.DENY_REQUEST) or greater
Deny the request, do not revoke.

throws

Exception	Result
UserExitException	Handled the same as a return value of 8, and exception text is displayed on the resulting web page

postRevoke method

```
public void postRevoke(java.lang.String domain,
                       RevokeCert revokeobject,
                       int RACFrc,
                       int RACFrsncode)
```

Purpose

Called after REVOKE requests

Parameters

domain
Domain name

revokeobject
RevokeCert object containing reason number and serial number

preQRecover method

```
public int preQRecover(java.lang.String domain,
                       QRecover qrecoverobject)
```

Purpose

Called before QRECOVER requests

Parameters

domain
Domain name

qrecoverobject
QRecover object containing recovery email, passphrase, and security responses

returns

Value	**Meaning**
0 (UserExit.SUCCESSFUL)	Continue with the request.
4 (UserExit.QRECOVER_PASSPHRASE_NOT_FOUND)	Cannot determine the passphrase.
8 (UserExit.DENY_REQUEST) or greater	Deny the request.

postQRecover method

```
public int postQRecover(java.lang.String domain,
                        QRecover qrecoverobject,
                        int RACFrc,
                        int RACFrsncode)
```

Purpose

Called after QRECOVER requests

Parameters

domain
Domain name

qrecoverobject
QRecover object containing recovery email, passphrase, security responses, and a QrecoverResultsList array containing data for certificates that matched the search criteria

Class ExportCert

```
public class ExportCert
extends java.lang.Object
```

ExportCert contains the parameters passed on an `R_PKIServ` EXPORT request.

Method summary	
`com.ibm.pki.rpkiserv.PkiCertificate`	**`getCertificate`**`()`
`java.lang.String`	**`getPassphrase`**`()`
`java.lang.String`	**`getTransactionid`**`()`
`void`	**`setPassphrase`**`(java.lang.String passphrase)`
`void`	**`setTransactionid`**`(java.lang.String transactionid)`

Class QRecover

Package: com.ibm.pki.web.exits

```
public class QRecover
extends java.lang.Object
```

QRecover contains the parameters passed on an `R_PKIServ` QRECOVER request.

Method summary	
`java.lang.String`	**`getPassphrase`**`()`
`com.ibm.pki.rpkiserv.QrecoverResultsList[]`	**`getQrecover_results`**`()`
`java.lang.String`	**`getRequestor`**`()`
`java.lang.String[]`	**`getSecurity_answers`**`()`
`void`	`(java.lang.String passphrase)`**`setPassphrase`**
`void`	**`setRequestor`**`(java.lang.String requestor)`
`void`	**`setSecurity_answers`**`(java.lang.String[] security_answers)`

Class RevokeCert

Package: com.ibm.pki.web.exits

```
public class RevokeCert
extends java.lang.Object
```

RevokeCert contains the parameters passed on an R_PKIServ REVOKE request.

Method summary	
int	**getReason**()
java.lang.String	**getSerial_number**()
void	**setReason**(int reason)
void	**setSerial_number**(java.lang.String serial_number)

Class UserExitException

Package: com.ibm.pki.web.exits

```
public class UserExitException
extends java.lang.Exception
```

An exception thrown from the UserExit class. The PKI Services function that detects this exception treats it the same as a return code greater than or equal to 8, and stops processing. Additionally, any exception text is displayed on the web page that reports the unsuccessful processing.

Constructors
UserExitException(java.lang.String message)
UserExitException(java.lang.Throwable cause)

Class CertPlist

Package: com.ibm.pki.rpkiserv

```
public class CertPlist
extends java.lang.Object
```

The CertPlist (certificate parameter list) is used to pass certificate information to the R_PKIServ callable service. It defines the following static constant strings, which should be used for the name parameter:

```
CERTPLIST_UNSTRUCTNAME
CERTPLIST_EMAILADDR
CERTPLIST_MAIL
CERTPLIST_TITLE
CERTPLIST_ORGUNIT
CERTPLIST_ORG
CERTPLIST_STREET
CERTPLIST_LOCALITY
CERTPLIST_STATEPROV
CERTPLIST_POSTALCODE
CERTPLIST_COUNTRY
CERTPLIST_KEYUSAGE
CERTPLIST_EXTKEYUSAGE
CERTPLIST_NOTBEFORE
CERTPLIST_NOTAFTER
CERTPLIST_ALTIPADDR
CERTPLIST_ALTURI
CERTPLIST_ALTEMAIL
CERTPLIST_ALTDOMAIN
CERTPLIST_ALTOTHER
CERTPLIST_CUSTOMEXT
CERTPLIST_NOTIFYEMAIL
```

```
CERTPLIST_PUBLICKEY
CERTPLIST_SIGNWITH
CERTPLIST_HOSTIDMAP
CERTPLIST_REQUESTOR
CERTPLIST_PASSPHRASE
CERTPLIST_USERID
CERTPLIST_LABEL
CERTPLIST_CERTPOLICIES
CERTPLIST_AUTHINFOACC
CERTPLIST_CRITICAL
CERTPLIST_SERIALNUMBER
CERTPLIST_DNQUALIFIER
CERTPLIST_UID
CERTPLIST_COMMONNAME
CERTPLIST_DOMAINNAME
CERTPLIST_EMAIL
CERTPLIST_CLIENTNAME
CERTPLIST_STARTDATE
CERTPLIST_ENDDATE
CERTPLIST_AUTORENEW
CERTPLIST_KEYSIZE
CERTPLIST_KEYALG
CERTPLIST_BUSINESSCAT
CERTPLIST_JURLOCALITY
CERTPLIST_JURSTATEPROV
CERTPLIST_JURCOUNTRY
```

Method summary	
`void`	**`addValue`**`(java.lang.String name, java.lang.String value)` Adds a value to the certPlist.
`void`	**`addValue`**`(java.lang.String name, java.lang.String[] values)` Adds values to the certPlist.
`java.util.Vector`	**`getNames`**`()` Gets the names of all name and value pairs in the certPlist.
`java.util.Vector`	**`getValues`**`(java.lang.String name)` Gets the vector of string values for this name in the certPlist.
`void`	**`removeAllValues`**`(java.lang.String name)` Deletes all of the values associated with a name in the certPlist.
`void`	**`removeValue`**`(java.lang.String name, java.lang.String value)` Deletes one of the values associated with a name in the certPlist.
`java.lang.String`	**`toString`**`()` Returns the string representation of this object.

Class PkiCertificate

Package: com.ibm.pki.rpkiserv

```
public class PkiCertificate
extends java.lang.Object
```

Contains a certificate generated by PKI Services. The PkiCertificate class defines the following static constant integers to be used in determining the type (format) of certificate exported:

Integer	Meaning
`PkiCertificate.PKIS_CERTIFICATE_TYPE_BASE64`	Certificate is BASE64-encoded
`PkiCertificate.PKIS_CERTIFICATE_TYPE_DER`	Certificate is DER-encoded
`PkiCertificate.PKIS_CERTIFICATE_TYPE_PKCS12`	Certificate is DER-encoded PKCS #12
`PkiCertificate.PKIS_CERTIFICATE_TYPE_PKCS7_CHAIN`	Certificate is DER-encoded PKCS #7 chain

Method summary	
`public java.lang.String`	**`getBase64Encoded`**`()` Certificate in Base64 format. Use this method if the certificate type is `PKIS_CERTIFICATE_TYPE_BASE64`.
`public byte[]`	**`getDerEncoded`**`()` Certificate in DER format. Use this method if the certificate type is one of the DER types: • `PKIS_CERTIFICATE_TYPE_DER` • `PKIS_CERTIFICATE_TYPE_DER_PKCS7_CHAIN` • `PKIS_CERTIFICATE_TYPE_DER_PKCS12`
`public int`	**`getType`**`()` Returns the certificate type.

Class QrecoverResultsList

Package: com.ibm.pki.rpkiserv

```
public class QrecoverResultsList
extends java.lang.Object
```

Contains QRECOVER results from `R_Pkiserv`.

Method summary	
`java.lang.String`	**`getIssuerDn`**`()` Returns the issuer's distinguished name.
`java.lang.String`	**`getKeyId`**`()` Returns the key Id.
`java.lang.String`	**`getPassPhrase`**`()` Returns the passphrase provided when the certificate request was made.
`java.lang.String`	**`getSerialNum`**`()` Returns the serial number.

Method summary	
`java.lang.String`	**`getSubjectDn`**`()` Returns the subject's distinguished name.
`java.lang.String`	**`getValidityDates`**`()` Returns the validity period in local time.

Class RpkiservException

Package: com.ibm.pki.rpkiserv

```
public class RpkiservException
extends java.lang.Exception
implements java.io.Serializable
```

Exception thrown by classes in the package com.ibm.pki.rpkiserv.

Constructor summary
`RpkiservException``()` Constructor to create an empty `RpkiservException` object.
`RpkiservException``(java.lang.String exceptionText)` Constructor to create an `RpkiservException` object with only exception text.

Part 4. Using PKI Services

This part explains how to use the PKI Services web pages and utilities.

- Chapter 18, "Using the end-user web pages," on page 375 shows the web pages for the end user and explains how to perform tasks such as requesting a certificate, obtaining the certificate, and renewing or revoking a certificate.
- Chapter 19, "Using the administration web pages," on page 405 shows the administration web pages and explains how to process certificate requests and certificates.
- Chapter 20, "Using PKI Services utilities," on page 431 explains using the PKI Services utilities.

  createcrls
  A UNIX program that initiates the certificate revocation list (CRL) creation task immediately.

  db2conv
  A UNIX program that converts the version format of object store and ICL Db2 tables.

  iclview
  A UNIX program that displays the entries in the issued certificate list (ICL).

  pkiprereg
  A UNIX program that creates Simple Certificate Enrollment Protocol (SCEP) preregistration records

  postcerts
  A UNIX program that creates posting objects for existing certificates. The PKI Services daemon later posts the certificates to an LDAP server.

  TemplateTool
  A Java program that validates an XML certificate template file and converts it to a text CGI template file, and converts a text CGI template file to an XML template file.

  vosview
  A UNIX program that displays the entries contained in the object store (request database).

  vsam2db2
  A UNIX program that converts data from the issued certificate list (ICL) and object store VSAM data sets into Db2 tables.

  vsamconv
  A UNIX program that converts the version format of object store and ICL VSAM data sets.

- Chapter 21, "Using the certificate management protocol (CMP) with PKI Services," on page 453 describes the support for CMP that PKI Services provides.

Chapter 18. Using the end-user web pages

This topic describes how the end user can use the PKI Services web pages.

Notes:

1. The PKI Services web pages in this topic might differ slightly from those on the web. If your installation customized the templates, the web pages in this topic might differ greatly from those you view on the web. Additionally, the pages might contain differences depending on the browser you are using. (This topic assumes you are using Internet Explorer.) If you need to see the exact content, view the pages on the web.
2. If you are using Internet Explorer on a Microsoft Windows system, you might need to set up the Windows system and Internet Explorer to work with PKI Services. For information about how to do this, see Appendix C, "Using the PKI Services web application with Internet Explorer on Windows systems," on page 691.

By default, the end user can perform the following tasks:

- Install a CA certificate into the browser.
- Request a new certificate.
- Pick up a previously requested certificate.
- Renew or revoke a previously issued browser certificate.
- Recover a certificate and private key, if PKI Services generated the keys for the certificate.
- Install the PKI Services ActiveX program needed to install a renewed certificate using the Internet Explorer browser.

Table 65 on page 375 lists the types of certificates you can request:

Table 65. Types of certificates you can request

Type of certificate	**Use**
One-year PKI SSL browser certificate	End-user client authentication using SSL
One-year PKI S/MIME browser certificate	Browser-based email encryption
One-year PKI generated key certificate	Generation of public and private keys by PKI Services
Two-year PKI browser certificate for authenticating to z/OS	End-user client authorization using SSL when logging on to z/OS
Two-year PKI Authenticode - code signing server certificate	Software signing
Two-year PKI Windows logon certificate	End-user client authentication for an Active Directory user logging in to a Windows desktop using a smart card
Two-year EV SSL server certificate	Extended Validation (EV) server certificate for a private organization
Five-year PKI SSL server certificate	SSL web server certification
Five-year PKI IPSEC server (firewall) certificate	Firewall server identification and key exchange
Five-year PKI intermediate CA server certificate	Subordinate (non-self-signed) certificate authority certification

Table 65. Types of certificates you can request (continued)

Type of certificate	Use
Five-year SCEP certificate	Creation of a preregistration record for certificate requestors. (Certificate requestors using Simple Certificate Enrollment Protocol (SCEP) must be preregistered.) Unlike other templates, this template is intended for administration use only.
Two-year EST certificate	Creation of a preregistration record for certificate requestor. (Certificate requestor using Enroll Over Secure Transport (EST) must be preregistered.) Unlike other templates, this template is intended for administration use only.
n-year PKI browser certificate for extensions demonstration	Demonstration of all extensions supported by PKI Services
One-year SAF browser certificate	End-user client authentication where the security product (RACF, not PKI Services) is the certificate provider **Note:** The certificate generated by this template cannot be managed by the PKI Services administrator.
One-year SAF server certificate	Web server SSL certification where the security product (RACF, not PKI Services) is the certificate provider **Note:** The certificate generated by this template cannot be managed by the PKI Services administrator.

Special consideration for using SAF templates:

The templates that control processing of the SAF certificates listed in Table 65 on page 375 perform only a subset of the function available natively in RACF through the RACDCERT TSO command or the ISPF panels. They are provided to enable a web interface for requesting certificates from RACF for browsers and off-platform servers. They are not intended to be a complete replacement for RACF certificate function.

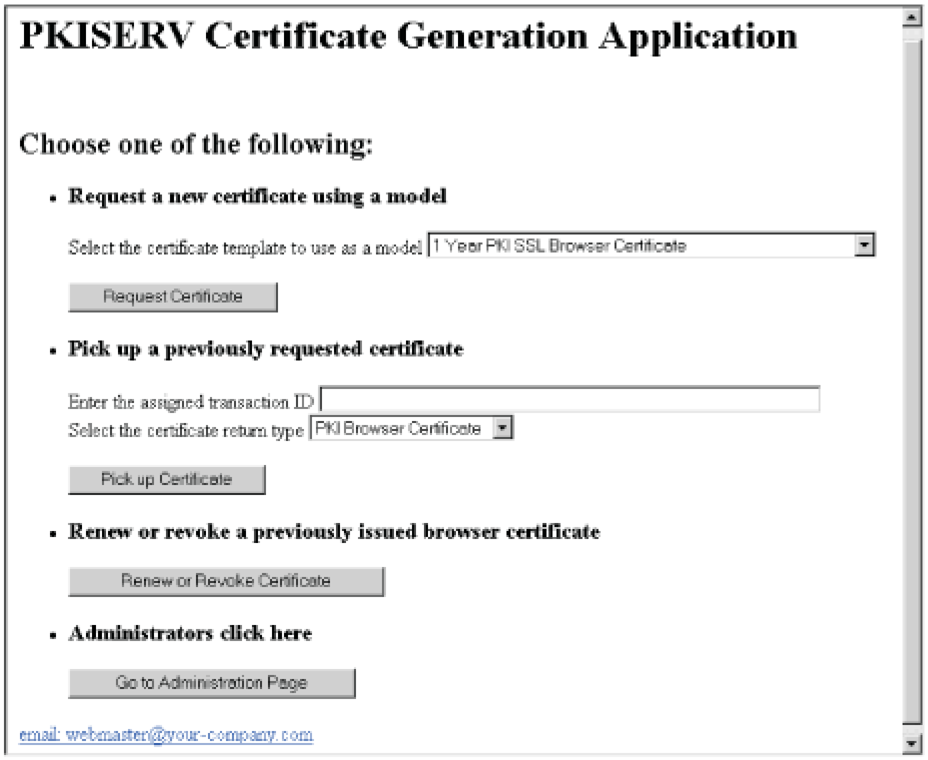
Restriction: If you want to generate a certificate for a server running on the local z/OS system (in other words, for a system using the RACF database where the signing certificate resides), do not use the "One-year SAF server certificate" template. Instead, use the RACDCERT TSO command or ISPF panels directly. Using the "One-year SAF server certificate" template might cause the loss of the private key if the authenticating user ID is not the same as the user ID specified when generating the certificate request in RACF.

Steps for accessing the end-user web pages

Perform the following preliminary steps to access the PKI Services web pages:

1. If this is the first time you access these web pages, you must install the CA certificate into your browser first.

Note: If you are using the Internet Explorer browser, you must explicitly select a store to place the certificate in. For more information, see "Installing the PKI Services CA certificate on a Microsoft Windows system" on page 694.



PKISERV Certificate Generation Application

Choose one of the following:

- **Request a new certificate using a model**

  Select the certificate template to use as a model 1 Year PKI SSL Browser Certificate

- **Pick up a previously requested certificate**

  Enter the assigned transaction ID

  Select the certificate return type PKI Browser Certificate

- **Renew or revoke a previously issued browser certificate**

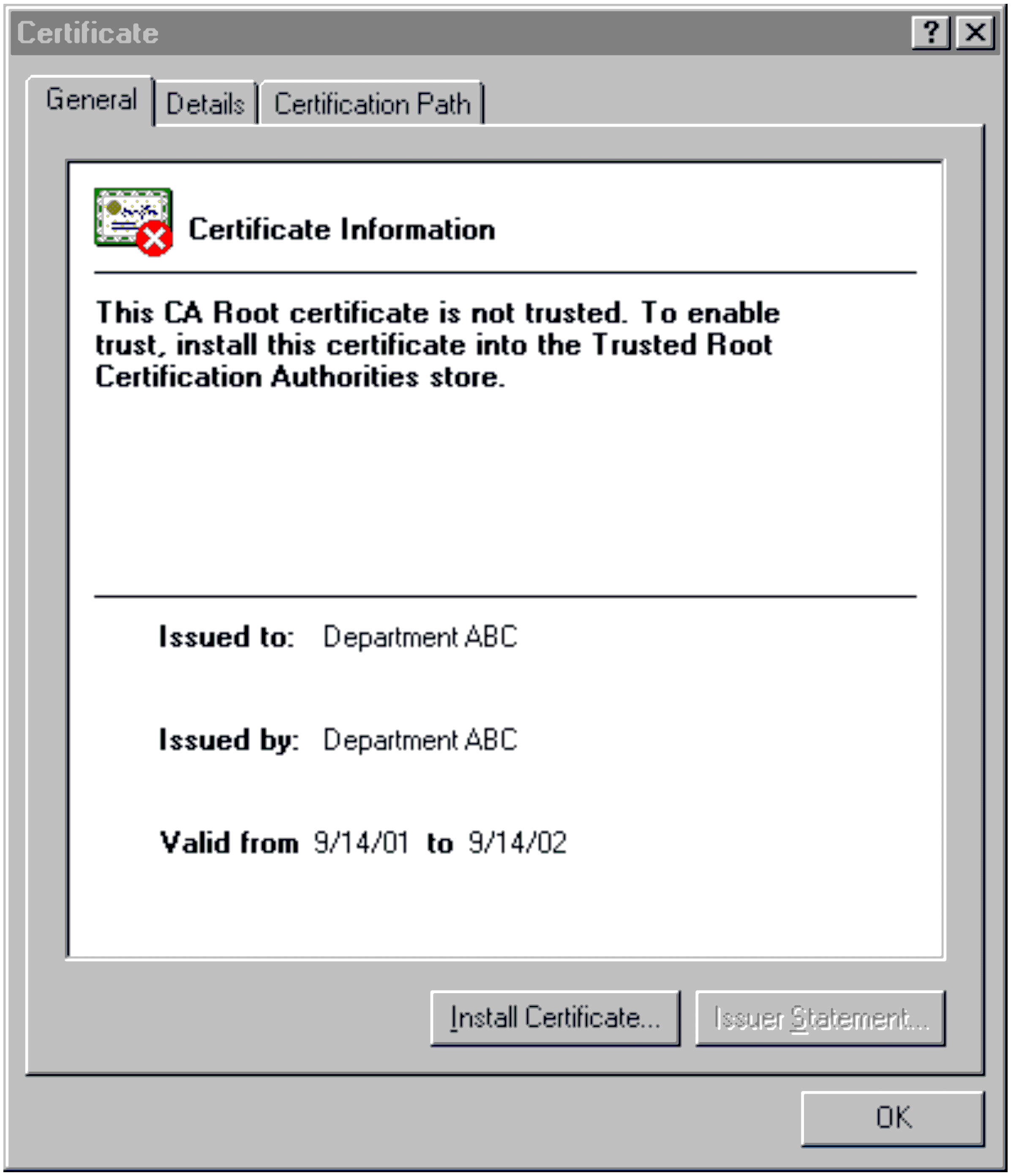
- **Administrators click here**

email: webmaster@your-company.com

Figure 42. PKI Services end-user home page for certificate generation

The following instructions are a sample of the directions to follow for installing the CA certificate on Internet Explorer:

a. After you click **Install the CA certificate**, a window labeled "File download" opens. Make sure that "Open this file from its current location" is selected (rather than "Save this file to disk"). Then click **OK**. Figure 43 on page 378 is an example of the window you might see, depending on the CA certificate you have installed.



Certificate Information

This CA Root certificate is not trusted. To enable trust, install this certificate into the Trusted Root Certification Authorities store.

Issued to: Department ABC

Issued by: Department ABC

Valid from 9/14/01 to 9/14/02

Figure 43. The certificate window for installing the CA certificate

b. Click **Install certificate**. (This initiates a series of windows in which you need to click **Next** and finally **Finish**, culminating in a window that says "The import was successful".)

Note: If you are using the Internet Explorer browser, you must explicitly select a store to place the certificate in. For more information, see “Installing the PKI Services CA certificate on a Microsoft Windows system” on page 694.

2. If you are using the Internet Explorer browser, now you can install the PKI Services ActiveX program that is used to install renewed certificates. For more information, see "Installing the PKI Services ActiveX program" on page 691. If you do not install it now, when you attempt to install a renewed certificate PKI Services checks whether you have the PKI Services ActiveX program installed. If it is not installed, PKI Services prompts you to install it.

   Note: Install the PKI ActiveX Control to renew certificates appears on the PKI Services home page only if you are using the Internet Explorer browser.

You are now ready to perform tasks, such as:

- Requesting a new certificate
- Picking up a previously requested certificate
- Renewing or revoking a previously issued browser certificate

Summary of fields

When you request certificates, you provide information for the fields in certificate request forms. The following table describes the fields in the end-user web pages:

Table 66. Summary of fields in end-user web pages

Field	**Description**
Certificate fields	
	Certificate fields related to Subject's Distinguished Name **Notes:** 1. The values for these fields are the relative distinguished names (RDNs) that are saved in the subject's distinguished name (DN) in the certificate. 2. For a server certificate, a base64-encoded PKCS #10 certificate request is required. If you specify one or more of these fields, the subject's distinguished name supplied in the PKCS #10 certificate request is ignored and only the fields you specify are in effect. For example, suppose that the subject's distinguished name specified in the PKCS #10 certificate request contains three RDNs - common name, organizational unit, and country. If you specify a value for organizational unit, you must also specify values for common name and country, even though you are not changing them. If you do not, these two RDNs have no values.
Business Category	The business category. This field is a text field of up to 64 characters. This field is intended for use in Extended Validation (EV) certificates.
Common name	Your name, such as John Smith. (You can use your first and last name, in that order.) This is a text field of up to 64 characters. For SSL servers, the common name is the server's fully qualified domain name, for example, `www.ibm.com`.
Country	The country where your organization is located. This is a 2-character text field.
Distinguished name qualifier	Specifies information to add to the subject distinguished name of an entry to make it unambiguous.

Table 66. Summary of fields in end-user web pages (continued)

Field	**Description**
Domain component	One component of a domain name associated with the subject distinguished name. For example, the domain name www.ibm.com is represented by 3 components: www, ibm, and com.
Email address	Email address with attribute EMAIL for the distinguished name. This is a text field of up to 64 characters.
Jurisdiction Country	The jurisdiction of incorporation country name. This field is a two-character text field. This field is intended for use in Extended Validation (EV) certificates.
Jurisdiction Location	The jurisdiction of incorporation locality name. This field is a text field of up to 64 characters. This field is intended for use in Extended Validation (EV) certificates.
Jurisdiction State or Province	The jurisdiction of incorporation state or province name. This field is a text field of up to 64 characters. This field is intended for use in Extended Validation (EV) certificates.
Locality	The city or municipality where your organization is located, such as Pittsburgh or Paris. This is a text field of up to 64 characters.
Mail	Email address with attribute MAIL for the distinguished name. This is a text field of up to 64 characters. **Restriction:** If you specify a value for this parameter and for `Notification e-mail address`, the two values *must* be the same.
Organization	The legally registered name (or trademark name, for example, IBM) of your organization. This is a text field of up to 64 characters.
Organizational unit	The name of your division or department. (There can be more than one organizational unit field on a request form. For example, one could be for your department and another for your division.) This is a text field of up to 64 characters.
Postal code	Your postal code or zip code. This is a text field of up to 64 characters.
Serial number	Serial number of the subject device. This is a text field of up to 64 characters.
State or Province	The state or province where your organization is located. Your registration policies determine whether you spell out the full name of the state or province or use an abbreviation. This is a text field of up to 64 characters.
Street	Your street address. This is a text field of up to 64 characters.
Title	Your job title. This is a text field of up to 64 characters.
Unstructured address	The unstructured address of the subject device.
Unstructured name	The unstructured name of the subject device.
User ID	The system login name associated with the subject distinguished name.
Certificate fields related to validity period	

Table 66. Summary of fields in end-user web pages (continued)

Field	Description
Not after (date)	A number of days, added to the current date after which the certificate expires. By default, you can select either one year or two years for the time at which the certificate expires.
Not before (date)	A number of days, added to the current date (by default, you can select either `0` or `30`), before which the certificate is not valid.
Certificate fields related to extensions	
Alternate domain name	Domain name for alternate name. This is the host name of the machine where a certificate is installed. This is a text field of up to 100 characters. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
Alternate email address	Email address for alternate name, including the `@` character and any periods (.). This is a text field of up to 100 characters. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
Alternate IP address	The IP address for the alternate name. This unique IP address specifies the location of each device or workstation on the Internet. PKI Services supports both IP version 4 and IP version 6 addresses. The IP address is a text field of up to 45 characters: • For IP version 4, the IP address is in dotted decimal format; for example, `9.67.97.103`. • For IP version 6, the IP address is divided into eight 16-bit hexadecimal blocks separated by colons. Leading zeros in each 16-bit field are optional, and successive fields of zeros can be represented by double colons, but only once; for example `1:2::3:4` is equivalent to `0001:0002:0000:0000:0000:0000:0003:0004`. • In a mixed IP version 4 and IP version 6 environment, the IP address can be expressed in the format *x:x:x:x:x:x:d.d.d.d,* where the *x* values are the hexadecimal values of the six high-order 16-bit pieces of the address, and the *d* values are the decimal values of the four low-order 8-bit pieces of the address in standard IP version 4 representation; for example, `0:0:0:0:0:0:ABCD:1.2.3.4`, or the equivalent value `::ABCD:1.2.3.4` **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.
Alternate other name	Additional identifier for the alternate name. See your PKI Services administrator for information about this field.
Alternate uniform resource identifier (URI)	Uniform resource identifier for the alternate name. This is a name or address referring to an Internet resource; a URL is one kind of uniform resource identifier. This is a text field of up to 100 characters. **Note:** The value is one of the list of subject's alternate names that is saved in the subject alternate name extension in the certificate.

Table 66. Summary of fields in end-user web pages (continued)

Field	Description
Extended key usage	This indicates the intended purpose of the certificate. Possible values are: **clientauth** Client side authentication **codesigning** Code signing **emailprotection** Email protection **mssmartcardlogon** Smart card logon for Microsoft Windows users **ocspsigning** OCSP response signing **serverauth** Server side authentication **timestamping** Digital timestamping **cmcca** CMC Certificate Authority **cmcra** CMC Registration Authority **cmcas** CMC Archive Server **pkinitkdc** PKINIT Kerberos KDC **pkinitclientauth** PKINIT Kerberos client
`HostIdMapping`	This is the user ID for authorization purposes in the format: `subject-id@host-name` **Example:** `DSmith@ibm.com` This is a text field of up to 100 characters.
Key usage	The intended purpose of the certificate. Each possible value is shown in Table 67 on page 384 with its intended purpose and possible PKIX bits.
Base64-encoded PKCS #10 certificate request	

Table 66. Summary of fields in end-user web pages (continued)

Field	Description

Base64-encoded PKCS #10 certificate request	(This is for server or device enrollment only.) You create a certificate request on behalf of another server (which could be a z/OS server or other type of server) or device for which you are requesting a certificate. You use software specific to that server to generate the PKCS #10 request before going to the PKI Services website. Save the request in a file. Then open the file in a text editor such as Windows Notepad and copy and paste the contents into the text box on the enrollment form. A text area of 70 columns and 12 rows is allocated for this certificate request. Here is an example of the certificate request:
	`-----BEGIN NEW CERTIFICATE REQUEST-----`
	`MIIBiDCB8gIBADAZMRcwFQYDVQQDEw5Kb2huIFEuIFB1YmxpYzCBnzANBgkqhkiG`
	`9w0BAQEFAAOBjQAwgYkCgYEAsCT1cJHAGPqi6OjAyL+xNbt8z5ngmvqO2V0O3oYu`
	`/mEnQtRM96e+2jbmDCRo5tWVklG4OYf9ZVB5biURMJFLztfa4AVdEVtun8DH2pwc`
	`wiNIZZcC1Zym5adurUmyDk64PgiiIPMQS/t0ttG4c5U8uWSK0b1J4V4f7ps+t1aG`
	`t+cCAwEAAaAwMC4GCSqGSIb3DQEJDjEhMB8wHQYDVR0OBBYEFAlKTovBBvnFqDAO`
	`1oIhtRinwRC9MA0GCSqGSIb3DQEBBQUAA4GBAIbCVpwYvppIX3HHmpKZPNY8Snsz`
	`AJrDsgAEH51WOIRGywhqKcLLxa9htoQai6cdc8RpFVTwk6UfdCOGxMn4aFb34Tk3`
	`5WYdzOiHXg8MhHiB3EruwdWs+S7Fv3JhU3FLwU6lFLfAjbVi+35iEWQymOR6mE5W`
	`CathprmGfKRsDE5E`
	`-----END NEW CERTIFICATE REQUEST-----`
	For a sample of the enrollment form showing the text box for a PKCS #10 request, see Figure 45 on page 388.

PKI Services internal use fields	
Challenge passphrase	This is the passphrase you entered when requesting a certificate. You type the same passphrase, exactly as you typed it on the request form. This is a case-sensitive text field of up to 32 characters.
KeySize	The size of the key pair (public key and private key) that you want PKI Services to generate for you.
Label	The label assigned to the requested certificate. This is a text field of up to 32 characters. This field applies only to SAF certificates.
Notification email address	Email address for notification purposes. This is a text field of up to 64 characters. **Note:** If you specify a value for this parameter and for `Mail`, the two values **must** be the same.
Passphrase	You decide this value when requesting a certificate (and must later supply this value when retrieving the certificate). You enter and then reenter this when requesting a certificate. This is a case-sensitive text field of up to 32 characters. (There is no minimum number of characters, and you can use any characters, but alphanumeric characters (`A–Z`, `a–z`, and `0–9`) are suggested.
Requestor's name	Your name (for tracking purposes). This can be in any format, for example, John Smith or John. J. Smith. This is a text field of up to 32 characters. **Note:** For a PKI generated key certificate, the requestor name needs to be in the form of an email address.
Transaction ID	This is assigned after you request your certificate. When it is displayed, you need to record this number. This is a text field of up to 56 characters.
Browser-specific fields	

Table 66. Summary of fields in end-user web pages (continued)

Field	Description
Cryptographic service provider	(This is for the Internet Explorer browser only.) The cryptographic service provider to generate your public/private key pair. You select a value from the drop-down list. Larger keys are more secure, but they also increase the time that is needed for connecting to a secure session.
Key protection	(This is for the Internet Explorer browser only.) This asks if you want to enable private key protection. (The drop-down choices are **Yes** and **No**.)
Key size	(This is for Mozilla-based browsers only.) This is the key size for your public/private key pair. Select a value from the drop-down list. Larger keys are more secure, but they also increase the time needed for connecting to a secure session.

Table 67. KeyUsage values and their intended purpose and possible PKIX bits

KeyUsage value	Intended purpose	PKIX bits
certsign	Certificate and CRL signing	`KeyCertSign` and `cRLSign`
crlsign	CRL signing	`cRLSign`
dataencrypt, dataencipherment, or dataenciph	Data encryption	`dataEncipherment`
digitalsig or digitalsignature	Authentication	`digitalSignature`
docsign or nonrepudiation	Document signing	`nonRepudiation`
handshake	Protocol handshaking (for example, SSL)	`digitalSignature` and `keyEncipherment`
keyagree or keyagreement	Key agreement	`keyAgreement`
keycertsign	Certificate signing	`keyCertSign`
keyencrypt, keyencipherment, or keyenciph	Key transport	`keyEncipherment`

Steps for requesting a new certificate

To request a new certificate, first go to the PKI Services home page. (See "#unique_66/unique_66_Connect_42_mainpage" on page 376.)

Perform the following steps to request a new certificate:

1. Click the down arrow to the right of the field beside `Request a new certificate using a model`. This displays a list of certificate templates from which you can select.

   For SCEP and EST preregistration: Do not follow these steps to request a SCEP or EST (preregistration) certificate template. Instead, go to "Steps for preregistering an SCEP or EST client" on page 403.

   The following list shows the certificate templates that PKI Services provides by default. This list might differ from the certificate templates your installation provides because your installation can customize the certificate templates and web pages.

   - One-year SAF server certificate
   - One-year SAF browser certificate
   - One-year PKI SSL browser certificate (See Figure 44 on page 386 to see a sample of this web page.)
   - One-year PKI SSL S/MIME browser certificate

- One-year PKI generated key certificate
- Two-year PKI browser certificate for authenticating to z/OS
- Two-year PKI Authenticode - code signing server certificate
- Two-year PKI Windows logon certificate
- Two-year EV SSL server certificate
- Five-year PKI SSL server certificate
- *n*-year PKI browser certificate for extensions demonstration
- Five-year SCEP certificate - Preregistration
- Two-year EST certificate - Preregistration
- Five-year PKI IPSEC server (firewall) certificate
- Five-year PKI intermediate CA server certificate

____

2. Click one of the items in the list. The drop-down list then collapses so that only the certificate you selected appears in the field and is highlighted.

____

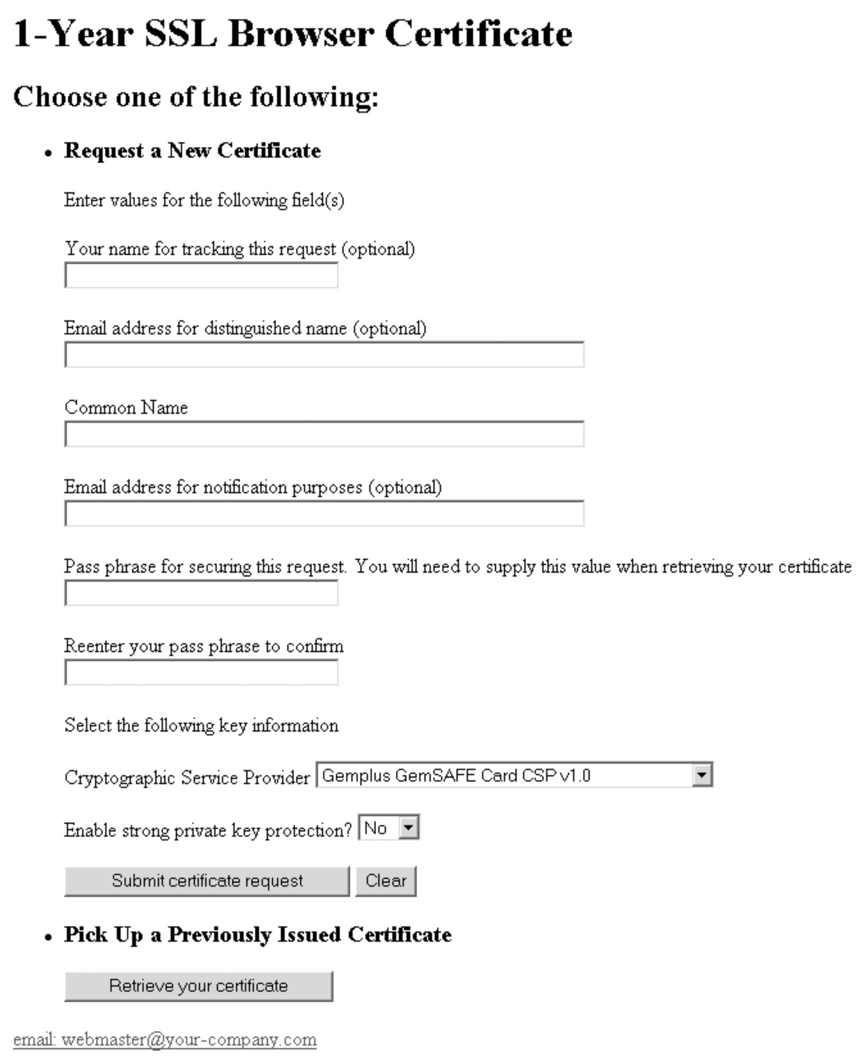
3. Click **Request certificate**. A form where you fill in information is displayed.

   Note: You might need to click through some additional panels specific to your browser (for example, clicking **Next** on a Mozilla-based browser or answering `Do you want to proceed?` on Internet Explorer) before the certificate request form appears.

____

4. Fill in the necessary information in the certificate request form.

   The form that appears depends on the certificate you are requesting and, in some instances, the fields that appear on the form depend on the browser you are using. **Example:** If you request a one-year SSL browser certificate, the form shown in Figure 44 on page 386 appears.



1-Year SSL Browser Certificate

Choose one of the following:

- **Request a New Certificate**

  Enter values for the following field(s)

  Your name for tracking this request (optional)

  Email address for distinguished name (optional)

  Common Name

  Email address for notification purposes (optional)

  Pass phrase for securing this request. You will need to supply this value when retrieving your certificate

  Reenter your pass phrase to confirm

  Select the following key information

  Cryptographic Service Provider: Gemplus GemSAFE Card CSP v1.0

  Enable strong private key protection? No

  Submit certificate request | Clear

- **Pick Up a Previously Issued Certificate**

  Retrieve your certificate

email: webmaster@your-company.com

Figure 44. One-year SSL browser certificate request form

Note: In the case of the one-year SSL browser certificate, fill in your common name. (See Table 66 on page 379 for descriptions of fields.) If you are using a Mozilla-based browser, select a key size from a drop-down list. Alternately, if you are using Internet Explorer, click the drop-down lists to select your cryptographic service provider and to specify whether to use strong private key protection.

____

5. If you are requesting a server or device certificate, you need to supply a base64-encoded PKCS #10 certificate request. Use software specific to that server to generate the PKCS #10 request before going to the PKI website. Paste the request into the web page as shown in Figure 45 on page 388.

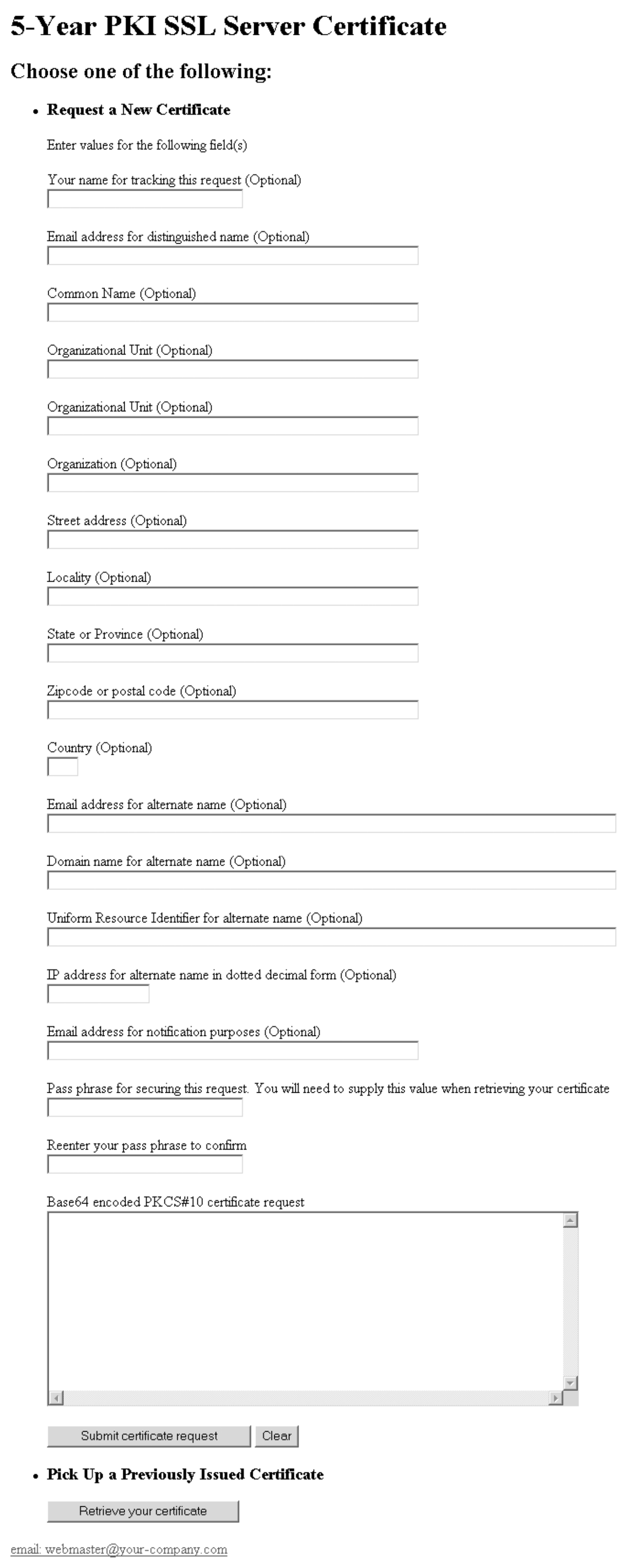
   For example, you could use the RACDCERT command to generate the PKCS #10 request. Assume that the server has the distinguished name OU=Inventory,O=XYZZY,C=US and a domain name xyzzy.com. This server runs on z/OS with the user ID INVSERV. First, generate a self-signed certificate for the server and assign the label "Inventory Server" to the certificate. The certificate is associated with the user ID that is associated with the server (INVSERV).

```
RACDCERT ID(INVSERV)
         GENCERT
         SUBJECTSDN(CN('xyzzy.com')
                    OU('Inventory')
                    O('XYZZY')
                    C('US'))
         WITHLABEL('Inventory Server')
```

Next, generate a PKCS #10 Base64-encoded certificate request based on the certificate you just created, and write the request to a data set.

```
RACDCERT ID(INVSERV)
         GENREQ(LABEL('Inventory Server'))
         DSN('WAIC.INVSERV.GENREQ')
```

Copy the PKCS #10 request from the data set WAIC.INVSERV.GENREQ and paste it into the field **Base64 encoded PKCS#10 certificate request**.



5-Year PKI SSL Server Certificate

Choose one of the following:

- **Request a New Certificate**

  Enter values for the following field(s)

  Your name for tracking this request (Optional)

  Email address for distinguished name (Optional)

  Common Name (Optional)

  Organizational Unit (Optional)

  Organizational Unit (Optional)

  Organization (Optional)

  Street address (Optional)

  Locality (Optional)

  State or Province (Optional)

  Zipcode or postal code (Optional)

  Country (Optional)

  Email address for alternate name (Optional)

  Domain name for alternate name (Optional)

  Uniform Resource Identifier for alternate name (Optional)

  IP address for alternate name in dotted decimal form (Optional)

  Email address for notification purposes (Optional)

  Pass phrase for securing this request. You will need to supply this value when retrieving your certificate

  Reenter your pass phrase to confirm

  Base64 encoded PKCS#10 certificate request

- **Pick Up a Previously Issued Certificate**

email: webmaster@your-company.com

Figure 45. Supplying the PKCS #10 certificate request for a server or device certificate

For server certificates where a base64-encoded PKCS #10 certificate request is supplied, specify one or more of the fields related to the subject's distinguished name only if you want to change the distinguished name supplied in the PKCS #10 certificate request. If you change one of these fields, the subject's distinguished name specified in the PKCS #10 certificate request is ignored and you must respecify the entire distinguished name (all fields). For a list of the fields related to the subject's distinguished name, see Table 66 on page 379.

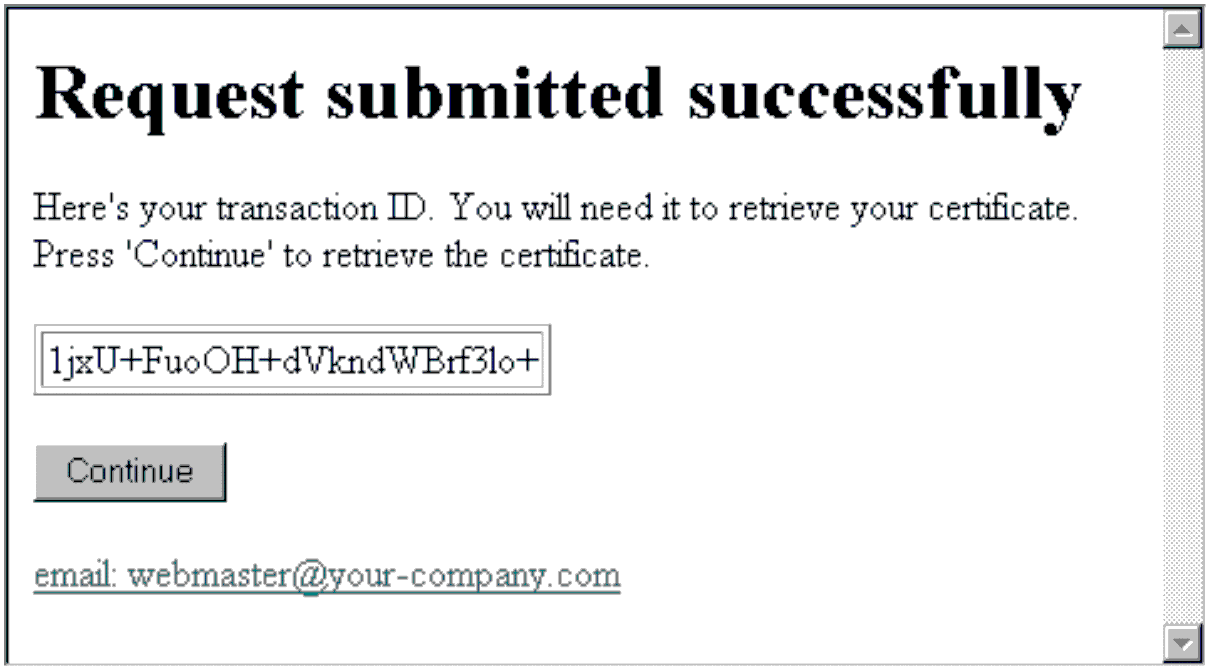
____

6. Fill in the passphrase on the certificate request form (twice). This is a value known only to you. Pick a value that you can easily remember because you need to supply the same passphrase when you pick up your certificate. Do not use a sensitive value such as your ATM pin or login password.

____

7. Fill in any optional information you want. When you are satisfied with the information you have entered, click **Submit certificate request**. If the request is successful, the results depend on the type of certificate you requested.

   - For all certificate types except one-year PKI generated key certificates, you see a page like the one shown in Figure 46 on page 389, which tells you your transaction ID.



Request submitted successfully

Here's your transaction ID. You will need it to retrieve your certificate. Press 'Continue' to retrieve the certificate.

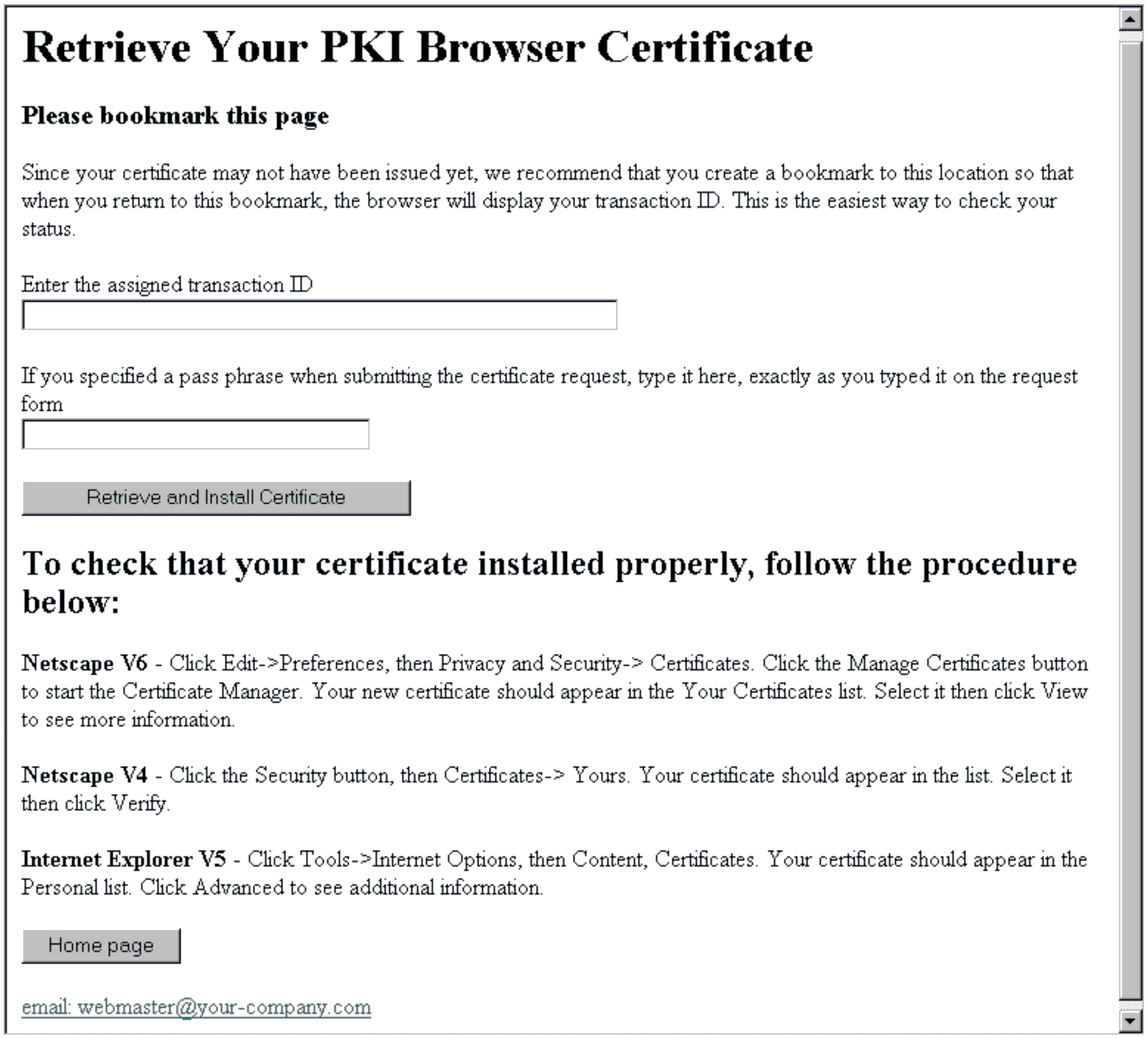
1jxU+FuoOH+dVkndWBrf3lo+

Figure 46. Successful request displays transaction ID

   a. Make a note of the transaction ID. (You can copy and paste the transaction ID to a file so that you have it for future reference, or you can write it in the following box. The reason for keeping a record of the transaction ID is that, depending on how you go to the web page to retrieve your certificate (see Figure 47 on page 390), you might have to fill in the transaction ID on that web page.)

```
Transaction ID:
```

   b. Click **Continue**. This displays the following web page:



Retrieve Your PKI Browser Certificate

Please bookmark this page

Since your certificate may not have been issued yet, we recommend that you create a bookmark to this location so that when you return to this bookmark, the browser will display your transaction ID. This is the easiest way to check your status.

Enter the assigned transaction ID

If you specified a pass phrase when submitting the certificate request, type it here, exactly as you typed it on the request form

To check that your certificate installed properly, follow the procedure below:

Netscape V6 - Click Edit->Preferences, then Privacy and Security-> Certificates. Click the Manage Certificates button to start the Certificate Manager. Your new certificate should appear in the Your Certificates list. Select it then click View to see more information.

Netscape V4 - Click the Security button, then Certificates-> Yours. Your certificate should appear in the list. Select it then click Verify.

Internet Explorer V5 - Click Tools->Internet Options, then Content, Certificates. Your certificate should appear in the Personal list. Click Advanced to see additional information.

email: webmaster@your-company.com

Figure 47. Web page to retrieve your certificate

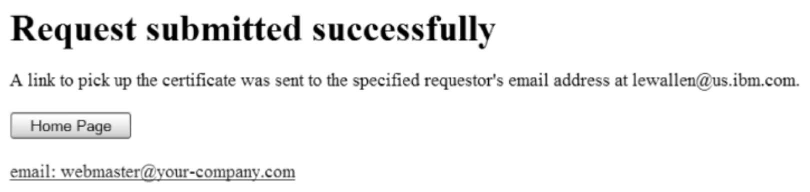
c. Bookmark this web page.

   Notes:

   1) After you submit the request for a certificate, your PKI Services administrator might need to approve the request before you can pick up your certificate. The amount of time that this takes can vary from a few minutes to a few days, depending on your installation. You bookmark this web page so that you can return to it at a later time.
   2) If your installation has enabled email notification and you supplied a valid email address when submitting your certificate request, then you receive an email message when your certificate is ready for pick-up or if PKI Services rejects your certificate request.

d. From this web page, you can start the steps to retrieve your certificate (see “Steps for retrieving your certificate from the bookmarked web page” on page 391) or you can return to the PKI Services home page (by clicking **Home**).

- For a one-year PKI generated key certificate, you see a page like the one shown in Figure 48 on page 391


Request submitted successfully

A link to pick up the certificate was sent to the specified requestor's email address at lewallen@us.ibm.com.

email: webmaster@your-company.com

Figure 48. Successful request for a one-year PKI generated key certificate

Unlike other types of certificates, this page does not show you the transaction ID for your certificate. Instead, PKI Services sends an email to the address you specified in the request. The email contains a link to the certificate.

____

Retrieving your certificate

For most certificate types, you can retrieve your certificate:

- From the web page you bookmarked in Step "7.c" on page 390. (This web page contains your transaction ID, so you do not have to enter it.) (See "Steps for retrieving your certificate from the bookmarked web page" on page 391.)
- From the PKI Services home page. (See "#unique_66/unique_66_Connect_42_mainpage" on page 376 and "Steps for retrieving your certificate from the PKI Services home page" on page 393.)

For a one-year PKI generated key certificate, you receive an email to notify you when your certificate is ready for retrieval. The email contains a link to the certificate. (See "Steps for retrieving a PKI generated key certificate" on page 393.)

If your company has enabled email notification for non-SAF certificates and you supplied a valid email address when submitting your certificate request, you receive an email to notify you when your certificate is ready for retrieval (or if your certificate request has been rejected).

Note: When a certificate is retrieved, it can be of different formats:

- If the keys are not generated by PKI Services, the returned format can be a single X.509 certificate or a chain of certificates, depending on the authority of the surrogate ID that does the EXPORT.
- If the PKI Services generated the keys for a certificate request, the returned format is a PKCS#12 package. The PKCS#12 package will by default contain the requested certificate, the private key, and the PKI Services CA certificate that is used to sign the requested certificate. However, the contents of the PKCS#12 package can be tailored using the PKCS#12 content configuration value in the `pkiserv.conf` file.

Steps for retrieving your certificate from the bookmarked web page

Perform the following steps to retrieve your certificate from the bookmarked web page:

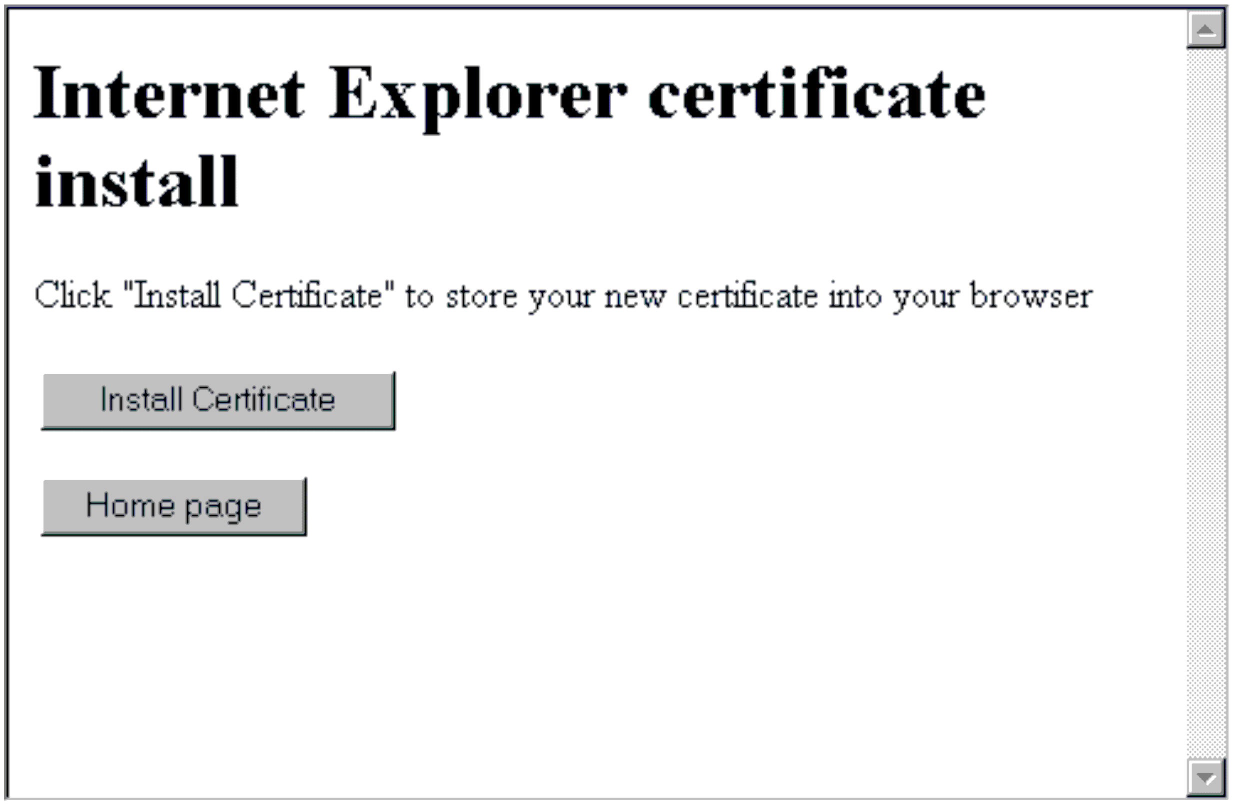
1. Go to the bookmarked web page. (See Figure 47 on page 390.)

   ____

2. If you entered a passphrase when requesting your certificate, enter the passphrase.

   ____

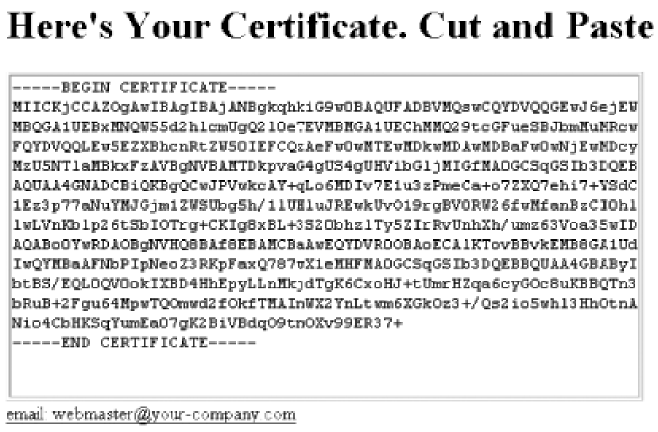
3. Click **Retrieve and install certificate**. If you are using a Mozilla-based browser, go to Step "5" on page 392. If you are using Internet Explorer and the retrieval of a certificate is successful, this displays the web page shown in Figure 49 on page 392. (This is for a browser certificate. For a server certificate, Figure 50 on page 392 shows an example of the web page.)



Internet Explorer certificate install

Click "Install Certificate" to store your new certificate into your browser

Figure 49. Browser certificate installation web page



Here's Your Certificate. Cut and Paste it to a File

```
-----BEGIN CERTIFICATE-----
MIICKjCCAZOgAwIBAgIBAjANBgkqhkiG9w0BAQUFADBVMQswCQYDVQQGEwJ6ejEW
MBQGA1UEBxMNQW55d2hlcmUgQ2l0eTEVMBMGA1UEChMMQ29tcGFueSBJbmMuMRcw
FQYDVQQLEw5EZXBhcnRtZW50IEFCQzAeFw0wMTEwMDkwMDAwMDBaFw0wNjEwMDcy
MzU5NTlaMBkxFzAVBgNVBAMTDkpvaG4gUS4gUHVibGljMIGfMA0GCSqGSIb3DQEB
AQUAA4GNADCBiQKBgQCwJPVwkcAY+qLo6MDIv7E1u3zPmeCa+o7ZXQ7ehi7+YSdC
1Ez3p77aNuYMJGjm1ZWSUbg5h/1lUHluJREwkUvO19rgBV0RW26fwMfanBzCIOhl
lwLVnKblp26tSbIOTrg+CKIg8xBL+3S2ObhzlTy5ZIrRvUnhXh/umz63Voa35wID
AQABo0YwRDAOBgNVHQ8BAf8EBAMCBaAwEQYDVR0OBAoECA1KTovBBvkEMB8GA1Ud
IwQYMBaAFNbPIpNeo23RKpFaxQ787vX1eMHFMA0GCSqGSIb3DQEBBQUAA4GBAByI
btBS/EQLOQVOokIXBD4HhEpyLLnMkjdTgK6CxoHJ+tUmrHZqa6cyGOc8uKBBQTn3
bRuB+2Fgu64MpwTQOmwd2fOkfTMAInWX2YnLtwm6XGkOz3+/Qs2io5wh13HhOtnA
Nio4CbHKSqYumEa07gK2BiVBdqO9tnOXv99ER37+
-----END CERTIFICATE-----
```

email: webmaster@your-company.com

Figure 50. Server certificate installation web page

---

4. Click **Install certificate**. If the certificate installs successfully, you get a popup window that says `Your new certificate installed successfully.`

---

5. Check that your certificate installed correctly:
   - For a Mozilla-based browser, navigate to **Options**, then **Advanced** → **Certificates** → **View Certificates** → **Your Certificates**. Your certificate should appear in the list. Select it and click Verify.
   - For Internet Explorer, click Tools → Internet Options, then Content, **Certificates**. Your certificate should appear in the Personal list. Click Advanced to see additional information.

Steps for retrieving your certificate from the PKI Services home page

Before you begin
To retrieve your certificate from the PKI Services home page, you must first know your transaction ID. You should have recorded this when your certificate request was successful. (See Figure 46 on page 389.)

Procedure

Perform the following steps to retrieve your certificate from the PKI Services home page:

1. Enter your transaction ID and select the certificate type using the drop-down. Then click **Pick up certificate** on the PKI Services home page. (See "#unique_66/unique_66_Connect_42_mainpage" on page 376.) This displays the web page that Figure 47 on page 390 shows.

   ____

2. Enter your passphrase (this is the challenge passphrase) if you specified one when requesting your certificate.

   ____

3. Click **Retrieve and install certificate**. If you are using a Mozilla-based browser, go to Step "5" on page 393. If you are using Internet Explorer and the retrieval of the certificate is successful, this displays the web page that Figure 49 on page 392 shows. (This is for a browser certificate. For a server certificate, Figure 50 on page 392 shows an example of the web page.)

   ____

4. Click **Install certificate**. If the certificate installs successfully, you get a popup window that says `Your new certificate installed successfully.`

   ____

5. Check that your certificate installed correctly:
   - For a Mozilla-based browser, click **Security**, then Certificates → Yours. Your certificate should appear in the list. Select it and click Verify.
   - For Internet Explorer, Click Tools → Internet Options, then Content, Certificates. Your certificate should appear in the Personal list. Click Advanced to see additional information.

   ____

Steps for retrieving a PKI generated key certificate

Perform the following steps to retrieve your PKI generated key certificate.

Before you begin
You need to have received an email indicating that your certificate is ready to be picked up.

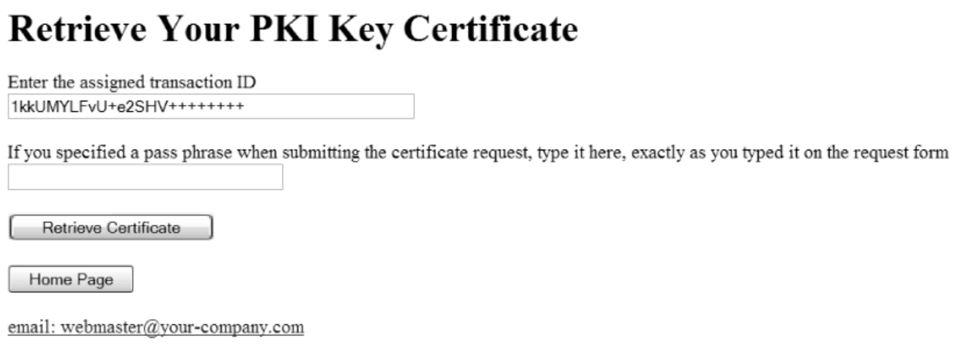
Procedure

1. The email informing you that your certificate is ready to be picked up appears similar to the one shown in Figure 51 on page 394. You have two alternatives:
   - Clink the link for your certificate in the email.
   - Copy the transaction ID from the email. Go to the PKI Services home page (see "#unique_66/unique_66_Connect_42_mainpage" on page 376). Paste the transaction ID into the field labeled "Enter the assigned transaction ID" and click **Pick up certificate**.

```
Attention - Please do not reply to this message as it was automatically
sent by a service machine.

Dear lewallen@us.ibm.com,

Thank you for choosing dime-o-cert PKI. The certificate you requested for
subject CN=Rocky,OU=Class 1 Internet Certificate CA,O=The Firm is now ready for pickup.

Please visit:
http://alps4027.pok.ibm.com/Customers/ssl-cgi-bin/caretrieve.rexx?TransactionId=1kkLpBZvA
%2
Bq%2F2SHV%2B%2B%2B%2B%2B%2B%2B%2B&Template=PKI+Key+Certificate
 to retrieve your certificate.

If that link does not work, try to go to
https://alps4027.pok.ibm.com/Customers/ssl-cgi/camain.rexx?
And enter the transaction ID listed below:
1kkLpBZvA+q/2SHV++++++++

You will need to input your passphrase that you entered when you submitted the request.
```

Figure 51. Email notification that your PKI generated key certificate is ready for pickup

2. The web page shown in Figure 52 on page 394 is displayed. Note that the transaction ID is filled in. Enter the passphrase that you entered when you submitted the certificate request, and click **Retrieve Certificate**.



Retrieve Your PKI Key Certificate

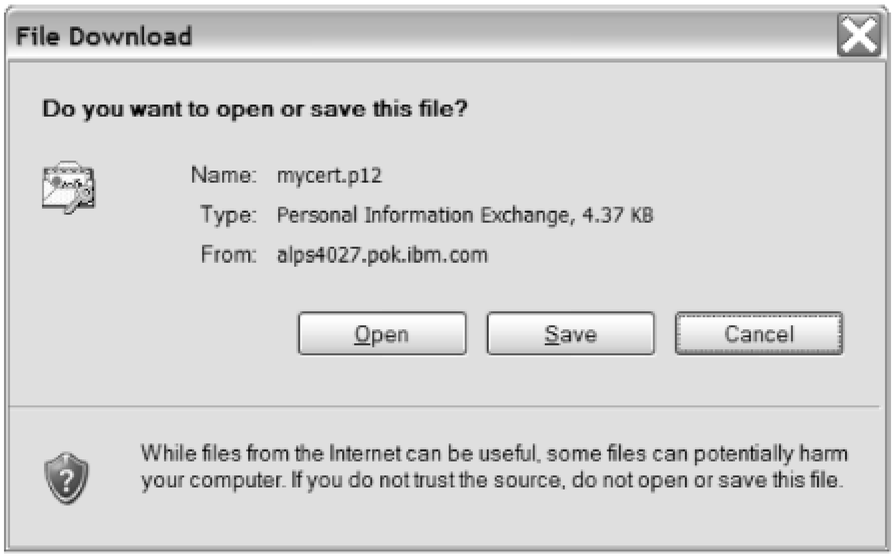
Enter the assigned transaction ID

1kkUMYLFvU+e2SHV++++++++

If you specified a pass phrase when submitting the certificate request, type it here, exactly as you typed it on the request form

Figure 52. Web page for retrieving a PKI generated key certificate

3. A window opens asking whether you want to open or save the PKCS #12 package containing the certificate and private key. This window is shown in Figure 53 on page 395.


File Download

Do you want to open or save this file?

Name: mycert.p12

Type: Personal Information Exchange, 4.37 KB

From: alps4027.pok.ibm.com

While files from the Internet can be useful, some files can potentially harm your computer. If you do not trust the source, do not open or save this file.

Figure 53. Window asking whether to open or save the PKCS #12 package

Click **Open** to invoke the Certificate Import Wizard to copy the certificate to a certificate store. Click **Save** to save the PKCS #12 package in a file.

____

Results

When you are done, you have retrieved your PKI generated key certificate and private key.

Steps for renewing a certificate

Note: For a PKI generated key certificate, if you have changed your email address since you originally requested the certificate, and the administrator has changed your email in the requester field in the certificate, you cannot renew the certificate.

Perform the following steps to renew a certificate:

1. On the PKI Services home page (see "#unique_66/unique_66_Connect_42_mainpage" on page 376), click **Renew or revoke certificate**. This displays a popup window with a list of certificates, such as the following figure shows:

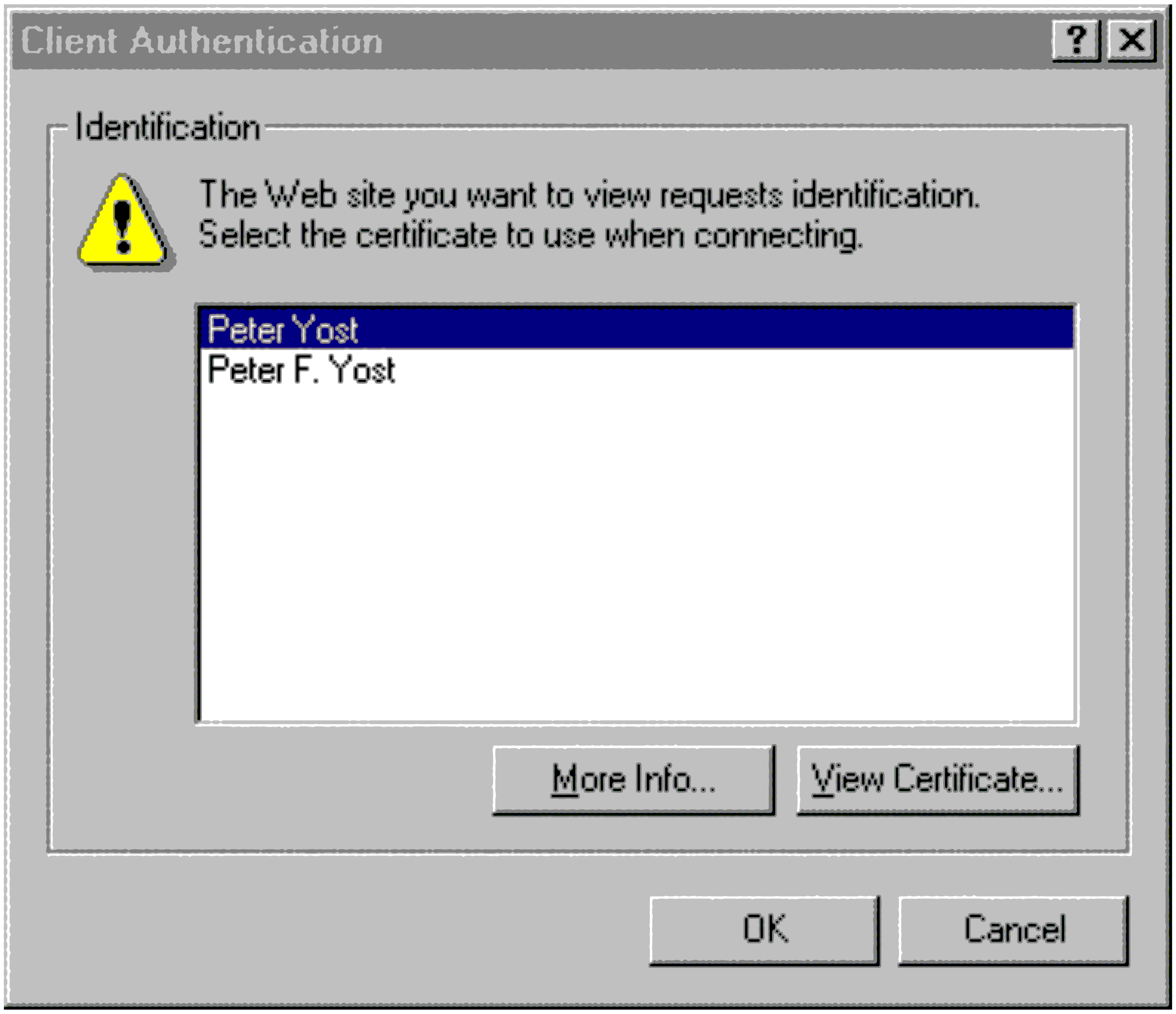
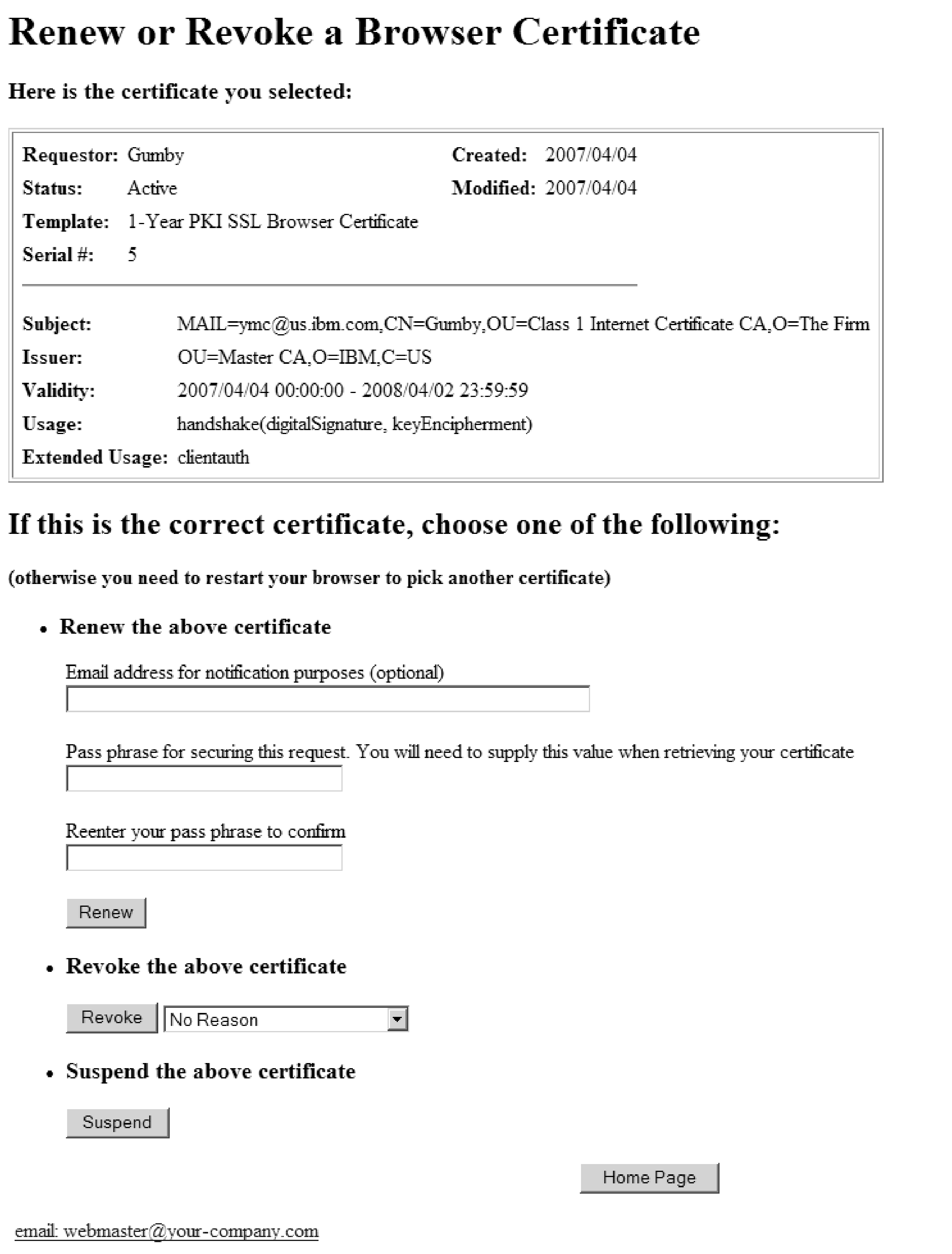

Client Authentication
Identification
The Web site you want to view requests identification.
Select the certificate to use when connecting.
Peter Yost
Peter F. Yost
More Info...
View Certificate...
OK
Cancel

Figure 54. Popup window listing certificates

2. The popup window might list more than one certificate. The certificates are listed by nickname in the order they are installed in the browser. Therefore, you might not be able to identify the PKI Services certificate you want to renew. Highlight the entry you think is the correct one and click **OK**. If the certificate you selected is one that PKI Services issued and it is not expired or revoked, Figure 55 on page 397 shows an example of the web page you might see, depending on the certificate you are renewing:



Renew or Revoke a Browser Certificate

Here is the certificate you selected:

Requestor:	Gumby	Created:	2007/04/04
Status:	Active	Modified:	2007/04/04
Template:	1-Year PKI SSL Browser Certificate		
Serial #:	5		

Subject:	MAIL=ymc@us.ibm.com,CN=Gumby,OU=Class 1 Internet Certificate CA,O=The Firm
Issuer:	OU=Master CA,O=IBM,C=US
Validity:	2007/04/04 00:00:00 - 2008/04/02 23:59:59
Usage:	handshake(digitalSignature, keyEncipherment)
Extended Usage:	clientauth

If this is the correct certificate, choose one of the following:

(otherwise you need to restart your browser to pick another certificate)

- Renew the above certificate

  Email address for notification purposes (optional)

  Pass phrase for securing this request. You will need to supply this value when retrieving your certificate

  Reenter your pass phrase to confirm

- Revoke the above certificate

- Suspend the above certificate

email: webmaster@your-company.com

Figure 55. Renew or revoke a certificate web page

Notes:

   a. If this is not the PKI Services certificate you want to renew, you need to close your browser (because the browser caches information) before again clicking **Renew or revoke certificate** as in Step "1" on page 395.
   b. If the certificate has the MAIL attribute in the subject's distinguished name, the value of NotifyEmail must match it.

   ____

3. Under the `Renew the above certificate` section, enter your passphrase in the two fields requesting it.

   ____

4. Click **Renew**.

   ____

5. If you are using Internet Explorer, and you do not have the appropriate PKI Services ActiveX program installed, PKI Services prompts you to install the ActiveX program. Follow the directions in the prompt. For more information, see "Installing the PKI Services ActiveX program" on page 691.
6. If the renewal request is successful, this displays a web page that says `Request submitted successfully` and displays the transaction ID. Click **Continue** on this web page.

   ____

7. This takes you the web page from which you retrieve your certificate. (See Figure 47 on page 390 for an example of this web page and "Steps for retrieving your certificate from the bookmarked web page" on page 391 for the directions to follow.)

   ____

Steps for revoking or suspending a certificate

Revoking or suspending a certificate means that you cannot continue to use the certificate. You might want to permanently *revoke* your certificate if you suspect your private key has been compromised. You might want to *suspend* (temporarily revoke) your certificate if you want to discontinue using it for a period of time (known as the suspension *grace period*).

If you suspend your certificate, the PKI administrator can *resume* (reactivate) the certificate, or permanently revoke it, if the certificate has not yet expired and the grace period has not elapsed. If the grace period has elapsed, the certificate is permanently revoked the next time the certificate revocation lists (CRLs) are issued.

Perform the following steps to revoke or suspend a certificate:

1. On the PKI Services home page (see "#unique_66/unique_66_Connect_42_mainpage" on page 376), click **Renew or revoke certificate**. This displays a popup window with a list of certificates, as in Figure 54 on page 396.

   ____

2. The popup window might list more than one certificate. The certificates are listed by nickname in the order they are installed in the browser. You might not be able to identify the PKI Services certificate you want to revoke or suspend. Highlight the entry that you think is the correct one and click **OK**. If the certificate you selected is one that PKI Services issued and it is not expired or revoked, this displays the "Renew or revoke a browser certificate" web page. (See "Steps for renewing a certificate" on page 395.)

   Note: If this is not the PKI Services certificate you want to revoke or suspend, you need to close your browser before again clicking **Renew or revoke certificate** as in Step "1" on page 398.

   ____

3. Make sure the certificate you want is the one described at the beginning of the web page. Click **Revoke** or **Suspend**. Note that when you revoke a certificate, you can click the drop-down list (of reasons) to select a reason, if you want.

   ____

4. This displays a web page that says `Request submitted successfully`. You can click **Home page** to return to the PKI Services home page.

   ____

When you are done: You can no longer use the certificate. If you suspended the certificate, contact your PKI administrator when you want to have it resumed.

Recovering a certificate whose keys were generated by PKI Services

If you request a PKI generated key certificate, PKI Services generates the public and private keys for the certificate and stores the certificate and keys in the ICSF token database (TKDS). If you lose the original certificate, PKI Services can recover the stored certificate and private key and return them to you in a PKCS #12 package.

Steps for recovering a certificate whose keys were generated by PKI Services

Before you begin

You need to know the email address you used when you requested the certificate. You also should know the passphrase that you entered on the certificate request. However, if you have forgotten the passphrase, and your company has implemented security questions, and you answered the security questions when you requested the certificate, you can provide those answers instead of the passphrase.

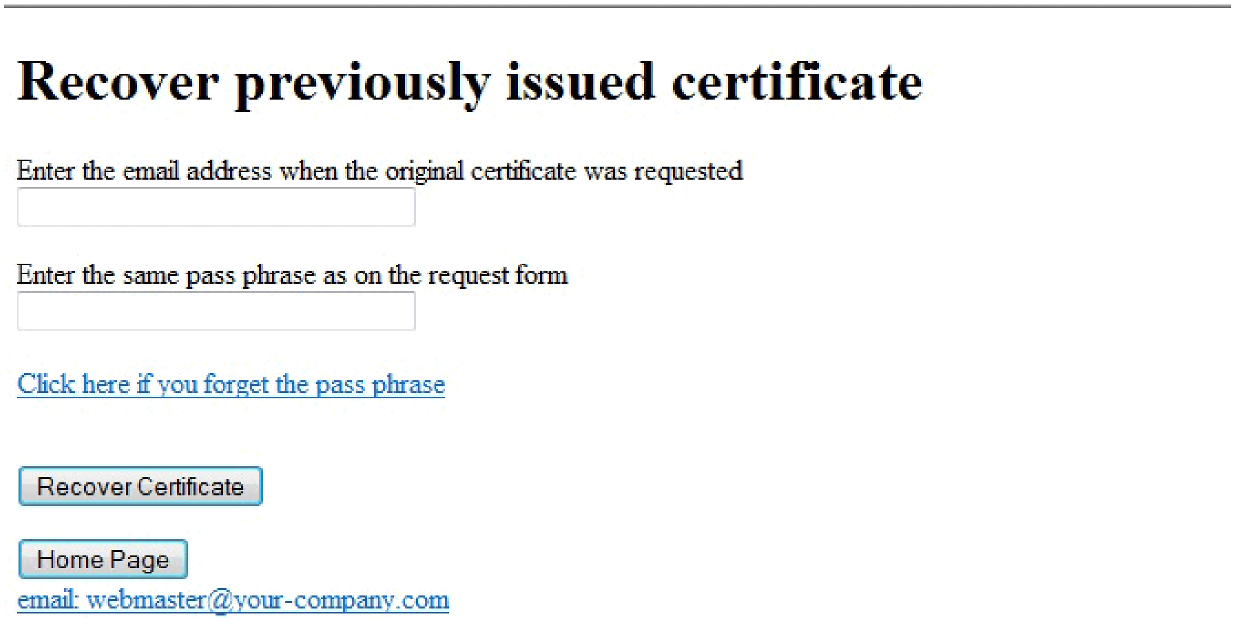
About this task

Perform the following steps to recover a certificate whose keys were generated by PKI Services.

Procedure

1. On the PKI Services home page (see “#unique_66/unique_66_Connect_42_mainpage” on page 376), click **Recover Certificate**.

   A window similar to the one shown in Figure 56 on page 400 opens.



Recover previously issued certificate

Enter the email address when the original certificate was requested

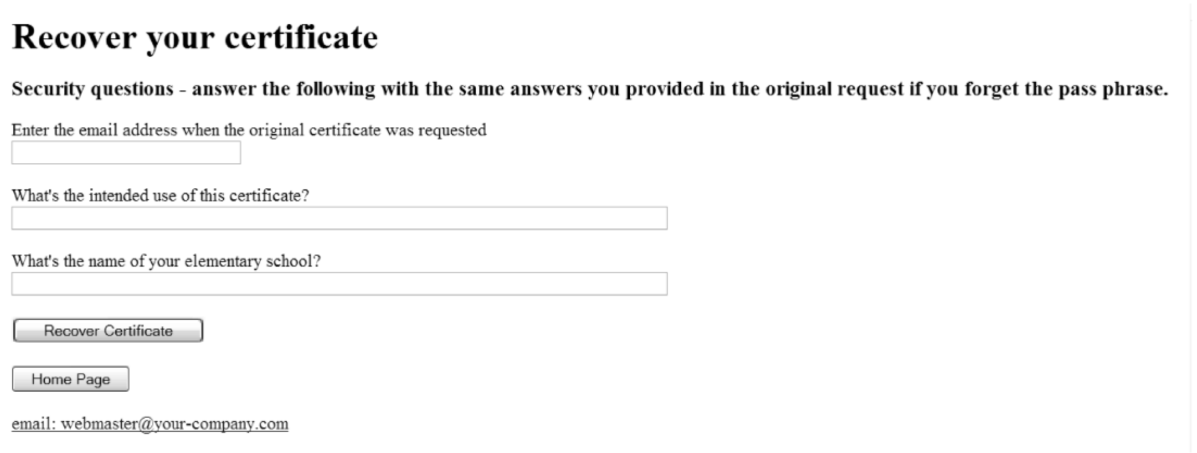
Enter the same pass phrase as on the request form

Click here if you forget the pass phrase

email: webmaster@your-company.com

Figure 56. web page to recover a certificate

2. On the "Recover previously issued certificate" window, take one of the following actions:
   a) If you remember the passphrase you used when you requested the certificate that you want to recover, enter the passphrase and the email address you used when you requested the certificate and click **Recover Certificate**.
   b) If you have forgotten the passphrase you used, click **Click here if you forget the pass phrase**. A web page similar to the one shown in Figure 57 on page 400 is displayed. Enter the email address you used when you requested the certificate and the answers to the security questions, and click **Recover Certificate**.



Recover your certificate

Security questions - answer the following with the same answers you provided in the original request if you forget the pass phrase.

Enter the email address when the original certificate was requested

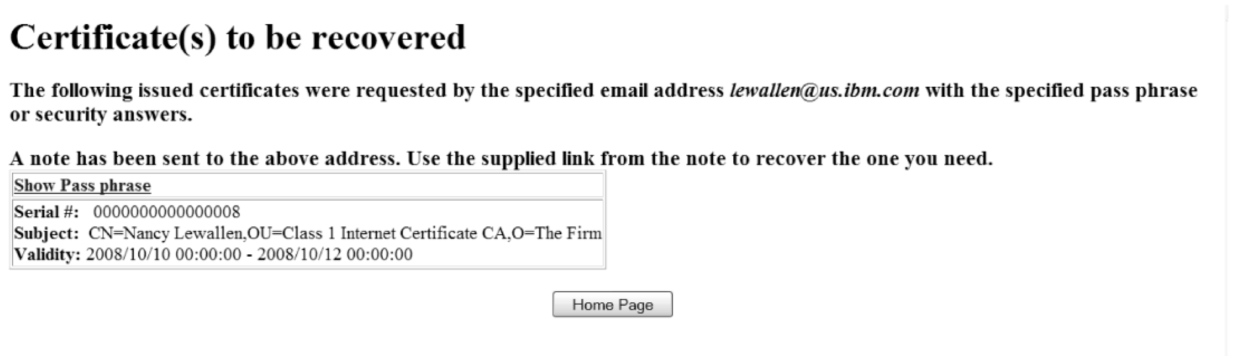
What's the intended use of this certificate?

What's the name of your elementary school?

email: webmaster@your-company.com

Figure 57. web page requesting answers to security questions when you have forgotten the passphrase

3. The web page shown in Figure 58 on page 401 is displayed listing the certificates that you can recover, and an email with links to those certificates is sent to your email address.


Certificate(s) to be recovered

The following issued certificates were requested by the specified email address *lewallen@us.ibm.com* with the specified pass phrase or security answers.

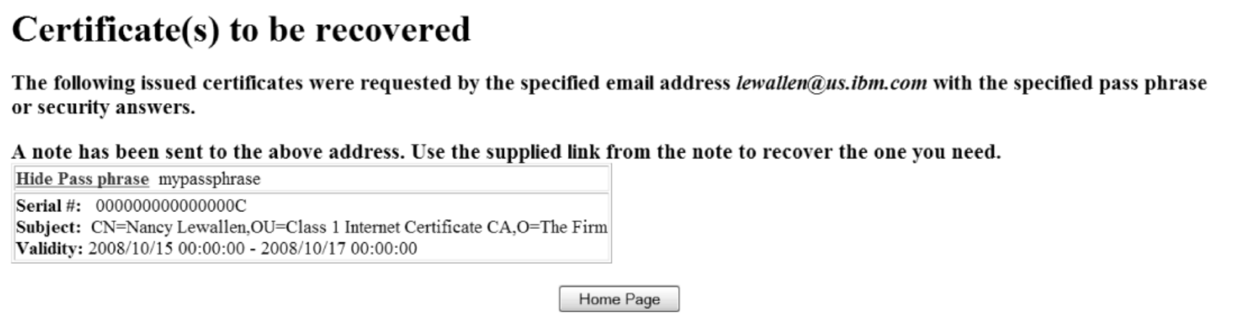
A note has been sent to the above address. Use the supplied link from the note to recover the one you need.

Show Pass phrase

Serial #: 0000000000000008
Subject: CN=Nancy Lewallen,OU=Class 1 Internet Certificate CA,O=The Firm
Validity: 2008/10/10 00:00:00 - 2008/10/12 00:00:00

Figure 58. web page listing certificates that can be recovered

Click **Show Pass phrase** to find out the pass phrase for the certificate you want to recover, if you have forgotten it. You need it to recover the certificate. The passphrase is displayed as shown in Figure 59 on page 401. Click **Hide Pass phrase** to hide the passphrase again.


Certificate(s) to be recovered

The following issued certificates were requested by the specified email address *lewallen@us.ibm.com* with the specified pass phrase or security answers.

A note has been sent to the above address. Use the supplied link from the note to recover the one you need.

Hide Pass phrase mypassphrase

Serial #: 000000000000000C
Subject: CN=Nancy Lewallen,OU=Class 1 Internet Certificate CA,O=The Firm
Validity: 2008/10/15 00:00:00 - 2008/10/17 00:00:00

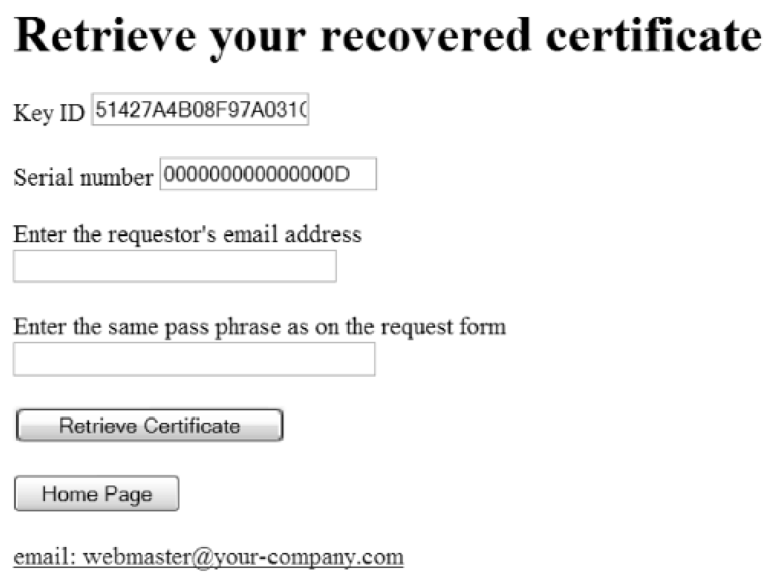
Figure 59. web page showing the passphrase for a certificate to be recovered

4. Open the email you were sent. Figure 60 on page 401 shows a sample email that lists one certificate eligible for recovery. Click the link for the certificate that you want to recover.

```
Attention - Please do not reply to this message as it was automatically sent by
a service machine.

Dear lewallen@us.ibm.com,

Here is a list of certificate(s) that satisfy your searching criteria for
recovery:
0000000000000008 : CN=Nancy Lewallen,OU=Class 1 Internet Certificate CA,O=The
Firm

Please choose the certificate you want and visit the corresponding link to
retrieve it (you can identify the certificate by the serial number from the
part of the link between '?' and '&')

https://www.dimeocert.com/Customers/ssl-cgi-in/caretrieve.rexx?
SerialNo=0000000000000008
&KeyID=2FBE1B1AC36F63C712AB6F5B829681549FD2095E

You need to input your pass phrase that you entered when you submitted the
request.
```

Figure 60. Sample email that lists certificates that can be recovered

5. The link takes you to the web page shown in Figure 61 on page 402.



Retrieve your recovered certificate

Key ID 51427A4B08F97A031(

Serial number 000000000000000D

Enter the requestor's email address

Enter the same pass phrase as on the request form

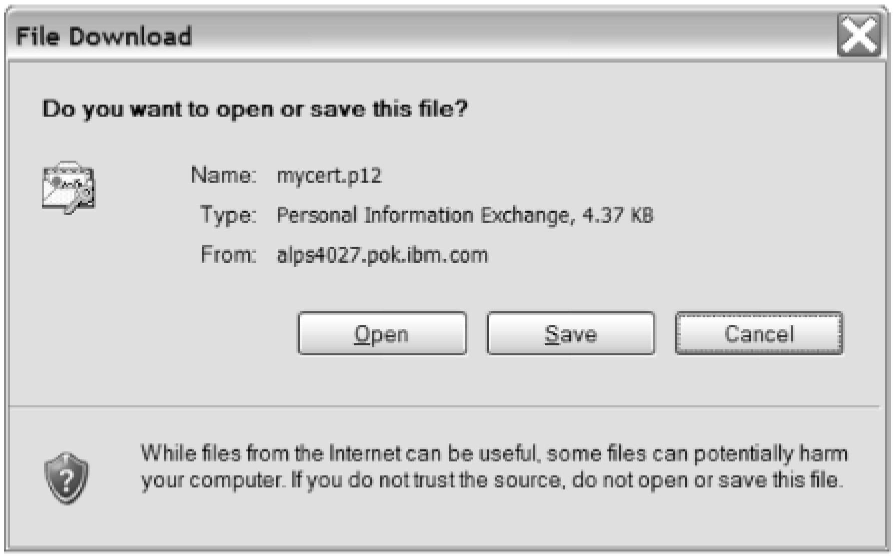
email: webmaster@your-company.com

Figure 61. web page to retrieve a recovered certificate

Fill in the email address and passphrase you used on the original certificate request, and click **Retrieve Certificate**.

____

6. A window opens asking whether you want to open or save the PKCS #12 package containing the certificate and private key. This window is shown in Figure 62 on page 402.



File Download

Do you want to open or save this file?

Name: mycert.p12

Type: Personal Information Exchange, 4.37 KB

From: alps4027.pok.ibm.com

While files from the Internet can be useful, some files can potentially harm your computer. If you do not trust the source, do not open or save this file.

Figure 62. Window asking whether to open or save the PKCS #12 package

Click **Open** to invoke the Certificate Import Wizard to copy the certificate to a certificate store. Click **Save** to save the PKCS #12 package in a file.

____

Results

When you are done, you have recovered your PKI generated key certificate.

Steps for preregistering an SCEP or EST client

These steps are performed by the PKI administrator using the end-user web pages. To preregister an SCEP or EST client, first go to the PKI Services home page. (See "#unique_66/unique_66_Connect_42_mainpage" on page 376.)

Perform the following steps to complete a preregistration request and preregister an SCEP or EST client:

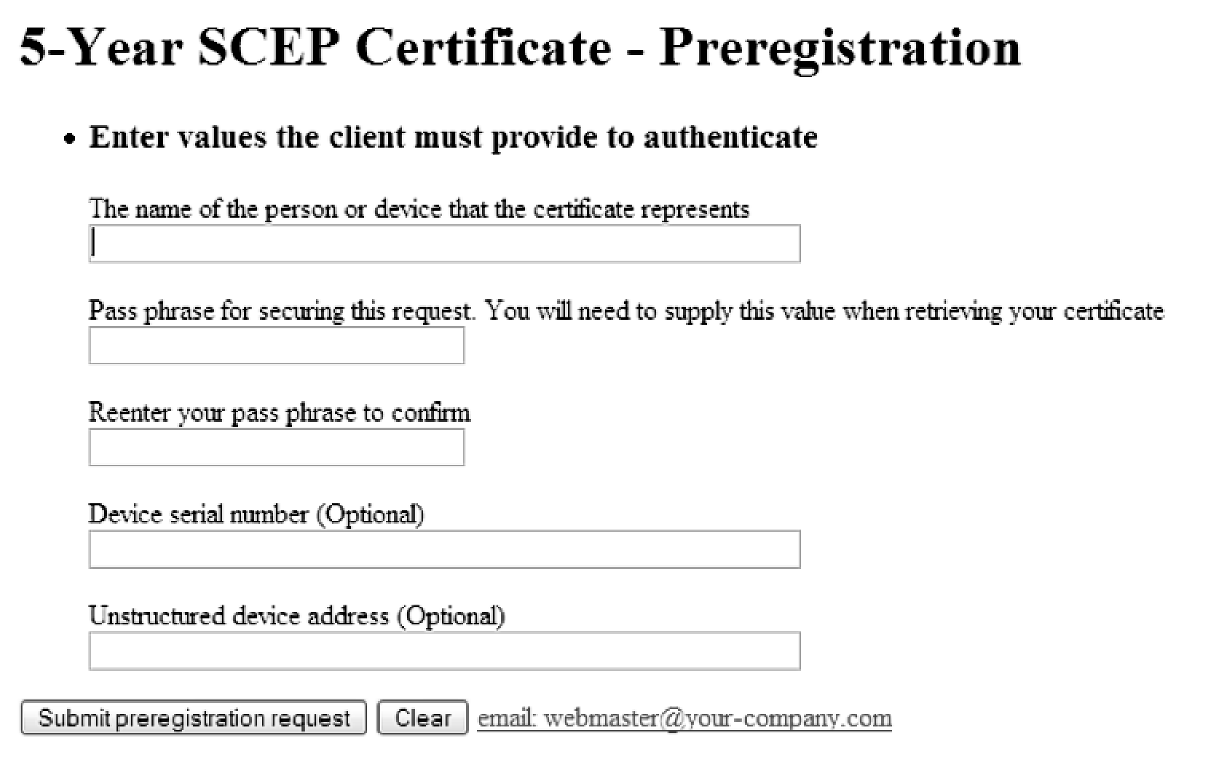
1. Click the down arrow to the right of the field beside `Request a new certificate using a model`. (This displays a list of certificate templates from which you can select.) Select `5-Year SCEP Certificate - Preregistration` from the list. The drop-down list then collapses so that only the preregistration template appears in the field and is highlighted.

   ____

2. Click **Request certificate**. A form where you fill in information is displayed.

   ____

3. The preregistration form appears. (See Figure 63 on page 403.) Fill in the necessary information in the preregistration form for this SCEP client. The form that appears depends how this template is customized for your CA domain.



5-Year SCEP Certificate - Preregistration

- Enter values the client must provide to authenticate

The name of the person or device that the certificate represents

Pass phrase for securing this request. You will need to supply this value when retrieving your certificate

Reenter your pass phrase to confirm

Device serial number (Optional)

Unstructured device address (Optional)

Submit preregistration request
Clear
email: webmaster@your-company.com

Figure 63. SCEP preregistration request form

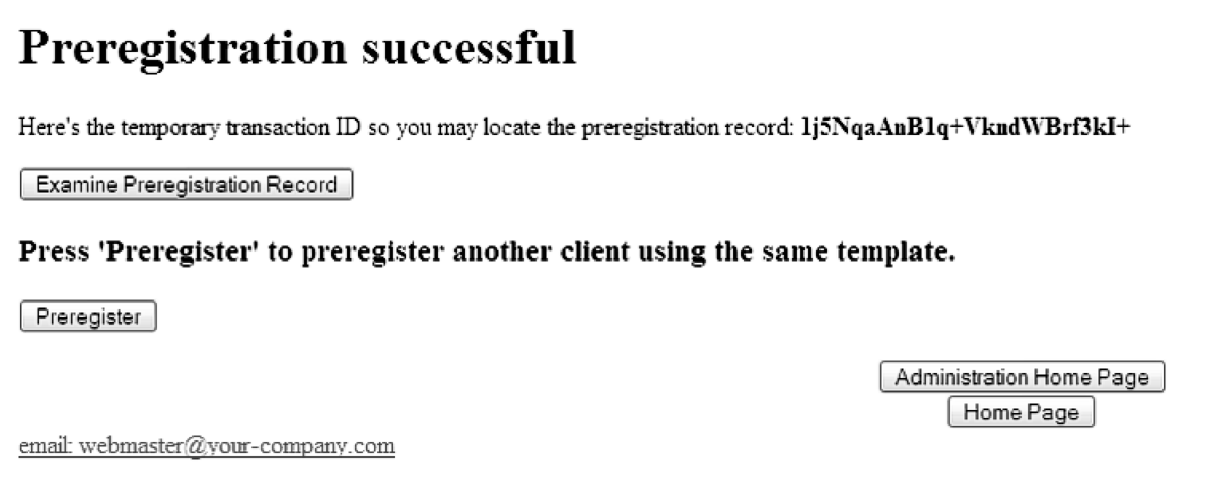
   ____

4. Fill in the passphrase on the certificate request form (twice). This SCEP or EST client must match this passphrase to successfully request a certificate. Do not use a sensitive value such as your ATM pin or login password.

   ____

5. Fill in any optional information as needed. This SCEP or EST client must match all subject and alternate name information you enter to successfully request a certificate. When you are satisfied with the information you have entered, click **Submit certificate request**.

If your preregistration request is successful, you see a page like the one shown in Figure 64 on page 404, which tells you your temporary transaction ID.



Preregistration successful

Here's the temporary transaction ID so you may locate the preregistration record: **1j5NqaAuB1q+VkndWBrf3kI+**

Examine Preregistration Record

Press 'Preregister' to preregister another client using the same template.

Preregister
Administration Home Page
Home Page
email: webmaster@your-company.com

Figure 64. Successful preregistration request

6. (Optional) Click **Examine Preregistration Record**.

7. (Optional) Click **Preregister** to preregister another client using the same template.

When you are done: You have successfully preregistered a SCEP or EST client. Return to the PKI Services home page (by clicking **Home Page**) or return to the administration home page (by clicking **Administration Home Page**).

Chapter 19. Using the administration web pages

This topic presents background information about certificate requests and certificates and explains how the administrator can use the administration web pages to perform the following tasks:

- Process a certificate request
  - Approve a request without making changes
  - Approve a request with changes
  - Reject a request
  - Delete a request
- Process a preregistration record
  - Delete a certificate request from a preregistered client

    (For information about preregistering clients, see "Steps for preregistering an SCEP or EST client" on page 403.)
- Process a certificate
  - Revoke a certificate
  - Suspend a certificate
  - Resume a certificate
  - Delete a certificate
- Perform searches for certificate requests, certificates, and preregistration records

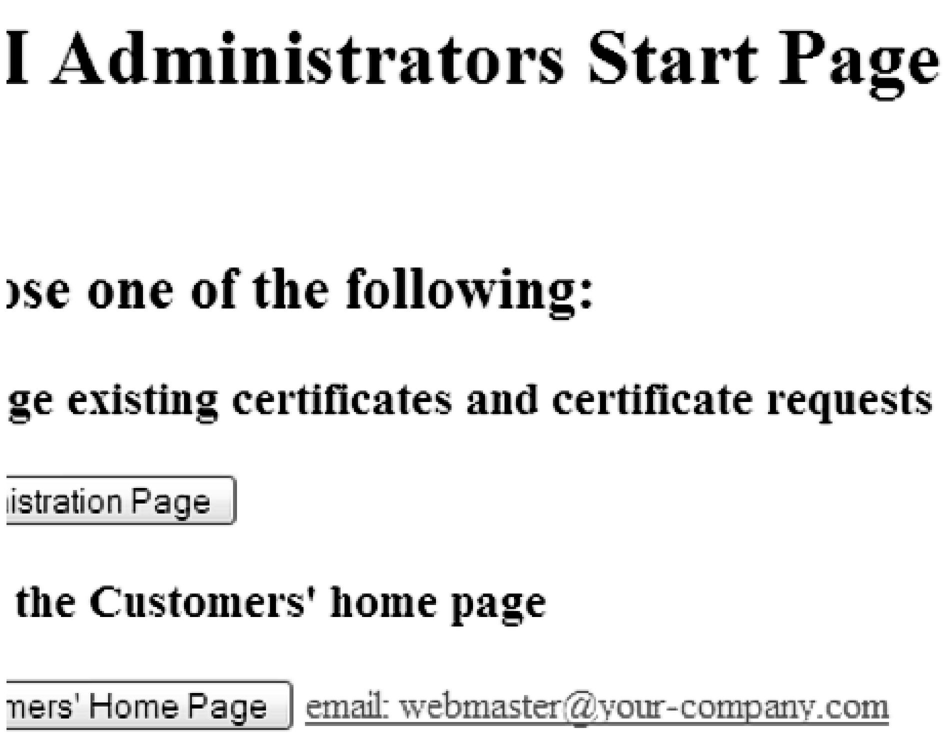
Note: The PKI Services web pages in this topic might differ slightly from those on the web. If you need to see the exact content, view the pages on the web. Additionally, the pages might contain differences depending on the browser you are using. This topic assumes that you are using Internet Explorer.

Steps for accessing the administration home page

Perform the following preliminary steps to access the administration home page:

1. If this is the first time you access these web pages, you must install the CA certificate into your browser first.

   Note: If you are using the Internet Explorer browser, you must explicitly select a store to place the certificate in. For more information, see "Installing the PKI Services CA certificate on a Microsoft Windows system" on page 694.



I Administrators Start Page

ɔse one of the following:

ge existing certificates and certificate requests

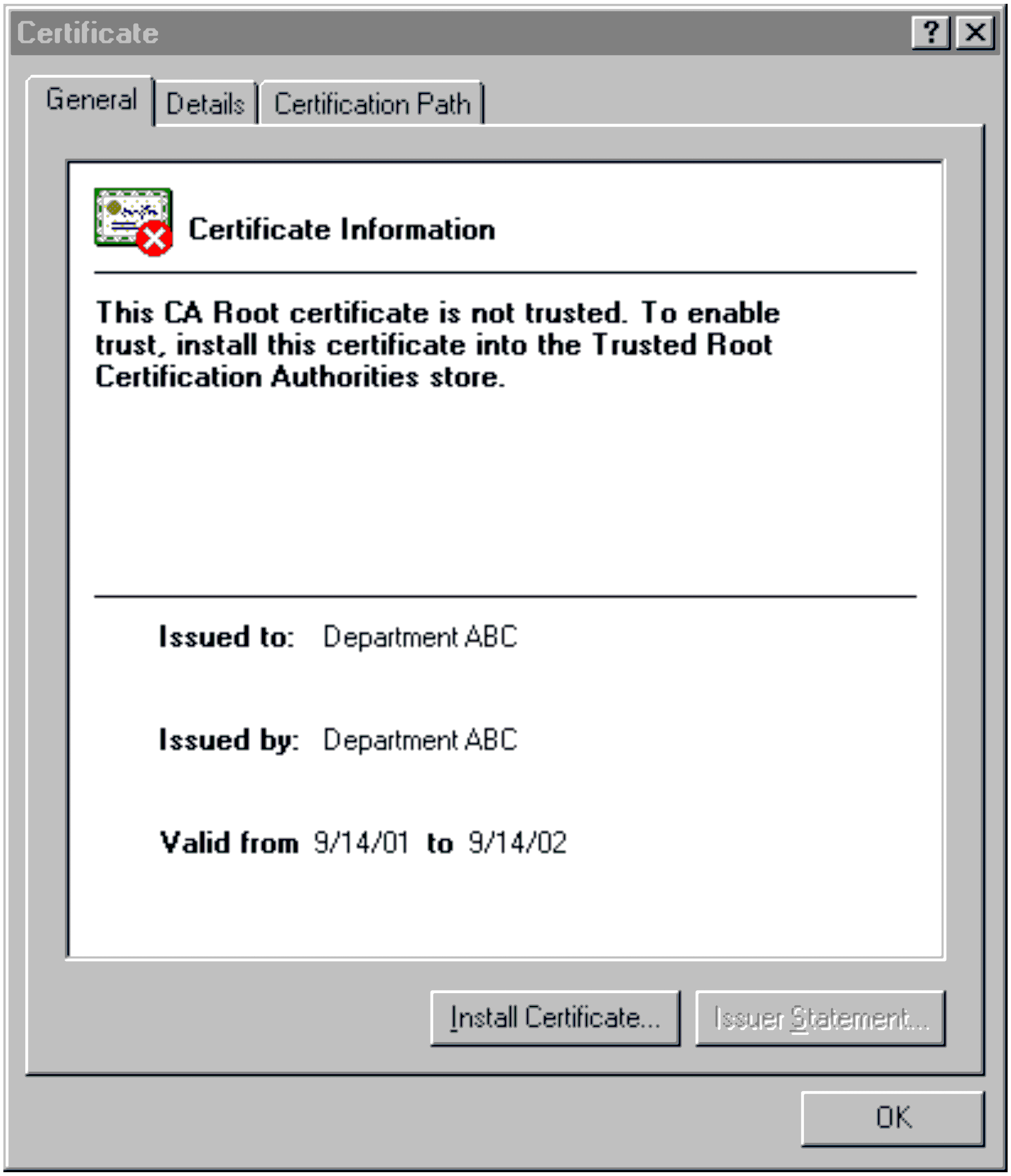
the Customers' home page

Figure 65. PKI Services administration start page

The following is a sample of the directions to follow for installing the CA certificate on Internet Explorer:

a. After you click the **Install the CA certificate** link, a popup window called "File download" appears. Make sure that the "Open this file from its current location" radio button is selected (rather than

"Save this file to disk"). Then click **OK**. The following is an example of the popup window you might see, depending on the CA certificate you have installed:



Certificate Information

This CA Root certificate is not trusted. To enable trust, install this certificate into the Trusted Root Certification Authorities store.

Issued to: Department ABC

Issued by: Department ABC

Valid from 9/14/01 to 9/14/02

Figure 66. The certificate popup window for installing the CA certificate

b. Click **Install certificate**. (This initiates a series of pop-ups in which you need to click **Next** and finally **Finish**, culminating in a popup window that says "The import was successful").

Note: You must explicitly select a store to place the certificate in. For more information, see "Installing the PKI Services CA certificate on a Microsoft Windows system" on page 694.

____

2. Click **Go to administration page**.

____

3. You are prompted to authenticate, as shown in the following figure. Provide the necessary information:

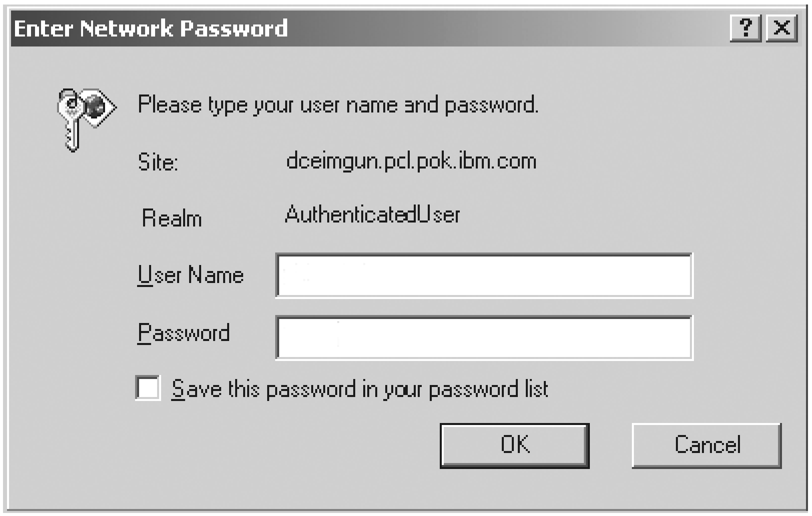

Enter Network Password
Please type your user name and password.
Site: dceimgun.pcl.pok.ibm.com
Realm AuthenticatedUser
User Name
Password
Save this password in your password list
OK
Cancel

Figure 67. Entering your user ID and password

   a. Fill in your z/OS user ID and password.
   b. If you want to eliminate having to reenter your user ID and password each time you access the administration pages, check the check box.
   c. Click **OK**.

      This calls up the "PKI Services Administration" web page. (See Figure 68 on page 411.)

   Notes:

   - Your web server programmer might provide you with an alternate URL for accessing the administration home page. You might also have to authenticate using a certificate instead of a user ID and password.
   - Your browser caches the authentication information that you provide. Therefore, if you need to change this information, you first must close all instances of your browser. Then open the browser and, when the panel shown in Figure 67 on page 408 appears, enter the correct information.

   ____

Fields in the administration web pages

When you process certificates requests and certificates, you provide information for various fields in the web pages. The following table describes the fields in the administration web pages:

Table 68. Summary of fields in the administration pages

Field	Description
Recent activity	This specifies a time range for searches. Possible values include: • Not selected • Within the past day • Within the past week • Within the past month • Within the past six months
Requestor name	The name of the person requesting the certificate, as it appears in the common name field of the certificate request form.

Table 68. Summary of fields in the administration pages (continued)

Field	Description
Serial number	PKI Services assigns this number to a certificate when you approve it.
Transaction ID	PKI Services assigns this number to a request when a user requests it. This is a text field of up to 56 characters.

Processing certificate requests

Before you can use the web page to process certificate requests, you need to understand the statuses of certificate requests and the actions you can perform on these certificate requests.

Status of certificate requests

Requests for certificates are kept in a request database while they are active. This is from the moment they are created until an event occurs that causes them to be deleted. The following table summarizes possible statuses. During the time period when a certificate request is active, it can have only one of the following statuses at a time:

Table 69. Statuses of certificate requests

Status	Meaning
`Pending Approval`	The request requires administrative approval. No action has been taken on the request yet. If the template that is used to create the certificate request specified an <ADMINNUM> value greater than 1, the request status might remain in Pending Approval state if the required number of individual administrator approvals have yet to be made for this request. If an Approve with Modifications is issued on a request that requires multiple approvals, all prior approvals are nullified and the current approval count for the request is set to 1.
`Approved`	The administrator explicitly approved the request or it was submitted as an auto-approved certificate request. The actual certificate might or might not have been created.
`Completed`	The certificate has been issued and the requestor has retrieved it. This is a final state.
`Preregistered`	The certificate request is from a preregistered Simple Certificate Enrollment Protocol (SCEP) or a Enrollment over Secure Protocol (EST) client.
`Rejected`	The administrator rejected the request, and the requestor has *not* been informed of this action (because the user has not tried to retrieve the certificate).
`Rejected, User notified`	The administrator rejected the request and the requestor has been informed of this action when attempting to retrieve the certificate. This is a final state.

A request is deleted from the request database when the administrator explicitly deletes it or when the request expires. This expiration time period is configurable and varies depending on whether the request was finalized or not.

Actions on certificate requests

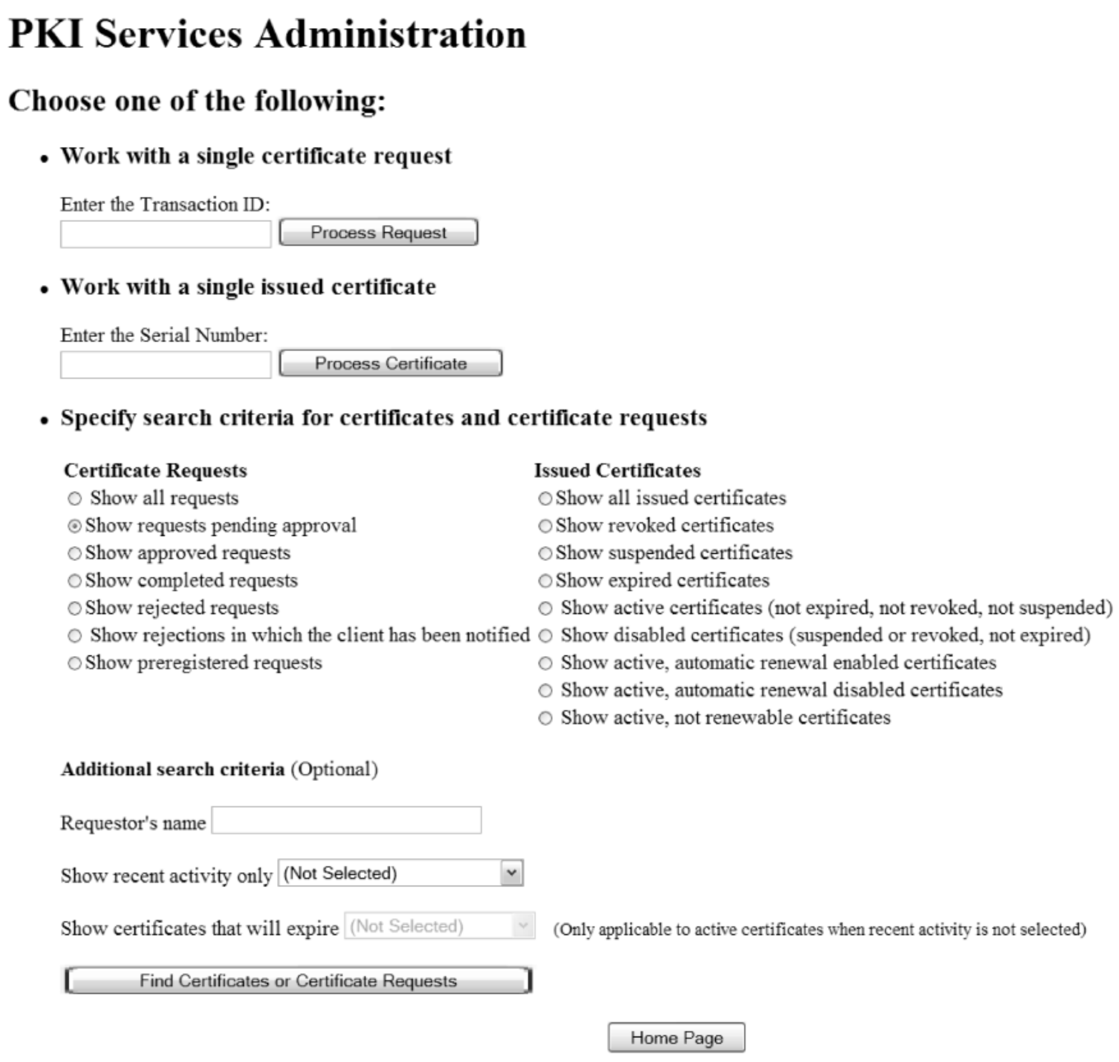
The following table summarizes actions on certificate requests and the required status for each of these actions:

Table 70. Summary of actions to perform on requests and required status

Action	Meaning	Required status of request
Approve	Approve the request in its current form. If this is the last required approval, queue a request to generate a certificate.	`Pending Approval`
Approve with modifications	Nullify any previous approvals for this request, alter the request as specified, and approve the request. If this request requires multiple approvals, the approval count is reset to 1, and the request remains in Pending Approval state.	`Pending Approval`
Reject	Reject the request and not allow for any future modifications of the request.	`Pending Approval`
Delete	Permanently remove the request.	All statuses (`Pending Approval`, `Approved`, `Completed`, `Preregistered`, `Rejected`, or `Rejected, Notified`)

Using the PKI Services administration home page

Figure 68 on page 411 shows the "PKI Services Administration" home page.



PKI Services Administration

Choose one of the following:

- Work with a single certificate request

  Enter the Transaction ID: [ ] Process Request

- Work with a single issued certificate

  Enter the Serial Number: [ ] Process Certificate

- Specify search criteria for certificates and certificate requests

Certificate Requests	Issued Certificates
○ Show all requests	○ Show all issued certificates
◉ Show requests pending approval	○ Show revoked certificates
○ Show approved requests	○ Show suspended certificates
○ Show completed requests	○ Show expired certificates
○ Show rejected requests	○ Show active certificates (not expired, not revoked, not suspended)
○ Show rejections in which the client has been notified	○ Show disabled certificates (suspended or revoked, not expired)
○ Show preregistered requests	○ Show active, automatic renewal enabled certificates
	○ Show active, automatic renewal disabled certificates
	○ Show active, not renewable certificates

Additional search criteria (Optional)

Requestor's name [ ]

Show recent activity only (Not Selected)

Show certificates that will expire (Not Selected) (Only applicable to active certificates when recent activity is not selected)

Find Certificates or Certificate Requests

Home Page

Figure 68. PKI Services administration home page

This web page allows you to:

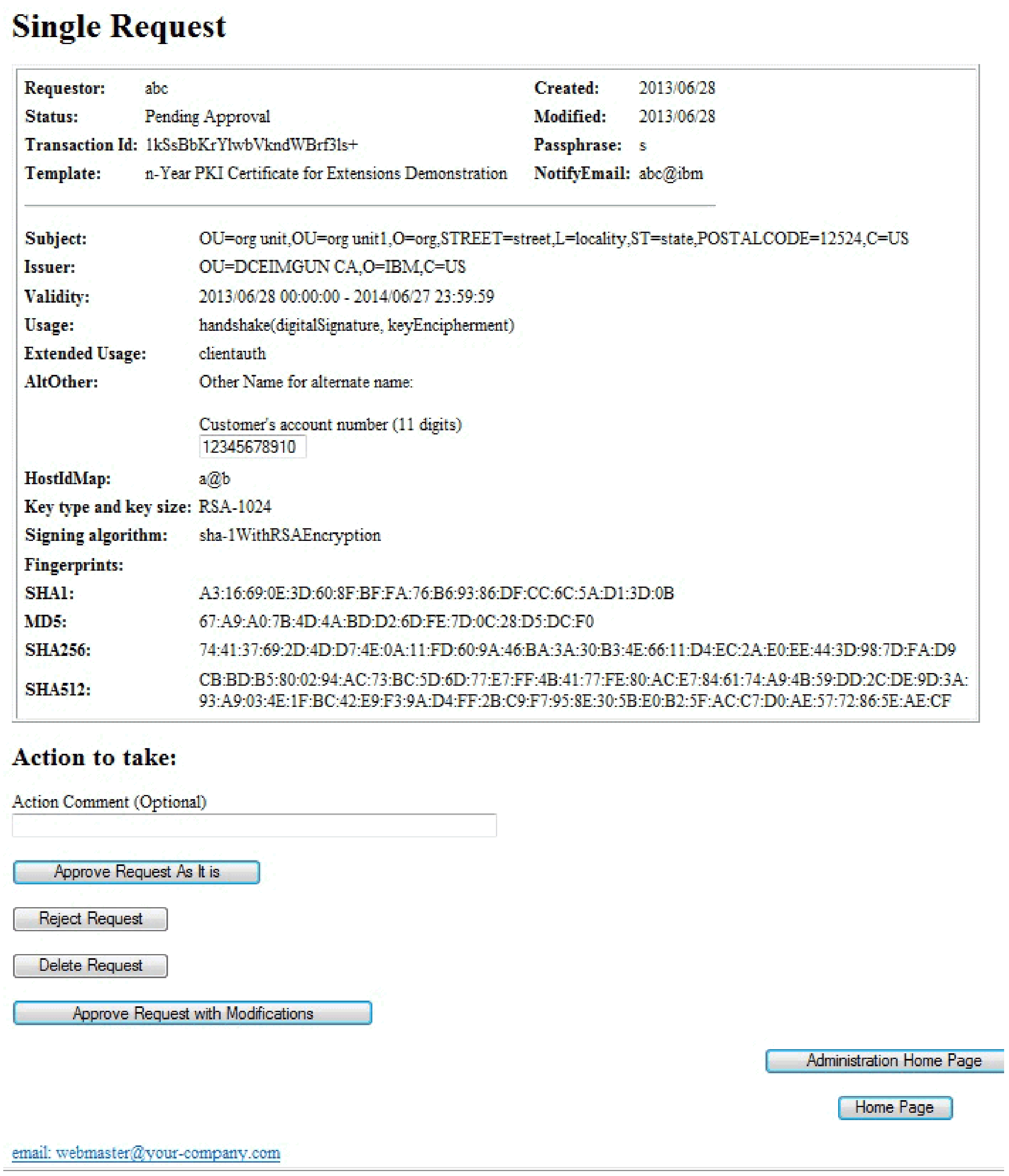
- Process a single certificate request (by specifying its transaction ID)
- Process a single certificate (by specifying its serial number)
- Search for groups of certificate requests or certificates by status and additional search criteria so that you can process them
- Search for certificates based on when they expire.

You can process a single certificate request or a single preregistered (certificate) request if you know its transaction ID. Otherwise, you can perform a search to display all certificate requests of a particular status.

Steps for processing a single request

To process a single request, perform the following steps:

1. On the PKI Services administration home page (see Figure 68 on page 411), enter the transaction ID in the field provided for it, and click **Process request**. This displays the "Single Request" web page shown in Figure 69 on page 412.



Single Request

Requestor:	abc	**Created:**	2013/06/28
Status:	Pending Approval	**Modified:**	2013/06/28
Transaction Id:	1kSsBbKrYlwbVkndWBrf3ls+	**Passphrase:**	s
Template:	n-Year PKI Certificate for Extensions Demonstration	**NotifyEmail:**	abc@ibm

Subject:	OU=org unit,OU=org unit1,O=org,STREET=street,L=locality,ST=state,POSTALCODE=12524,C=US
Issuer:	OU=DCEIMGUN CA,O=IBM,C=US
Validity:	2013/06/28 00:00:00 - 2014/06/27 23:59:59
Usage:	handshake(digitalSignature, keyEncipherment)
Extended Usage:	clientauth
AltOther:	Other Name for alternate name: Customer's account number (11 digits) 12345678910
HostIdMap:	a@b
Key type and key size:	RSA-1024
Signing algorithm:	sha-1WithRSAEncryption
Fingerprints:	
SHA1:	A3:16:69:0E:3D:60:8F:BF:FA:76:B6:93:86:DF:CC:6C:5A:D1:3D:0B
MD5:	67:A9:A0:7B:4D:4A:BD:D2:6D:FE:7D:0C:28:D5:DC:F0
SHA256:	74:41:37:69:2D:4D:D7:4E:0A:11:FD:60:9A:46:BA:3A:30:B3:4E:66:11:D4:EC:2A:E0:EE:44:3D:98:7D:FA:D9
SHA512:	CB:BD:B5:80:02:94:AC:73:BC:5D:6D:77:E7:FF:4B:41:77:FE:80:AC:E7:84:61:74:A9:4B:59:DD:2C:DE:9D:3A:93:A9:03:4E:1F:BC:42:E9:F3:9A:D4:FF:2B:C9:F7:95:8E:30:5B:E0:B2:5F:AC:C7:D0:AE:57:72:86:5E:AE:CF

Action to take:

Action Comment (Optional)

Figure 69. Single request approval web page

___

2. Make sure that the request is the correct one by reviewing the information in the top part of the web page.

   Guideline: If this is a preregistered certificate request, examine the *fingerprints* (the SHA1, SHA256, SHA512, and MD5 hashes) and contact the certificate requester to confirm that the fingerprints in this received request match the fingerprints in the original request. These actions ensure the integrity of the request. (The requester can use the SCEP client software to display the fingerprints saved for the original request.)

Note: If PKI Services is running in FIPS mode (the value specified by _PKISERV_FIPS_LEVEL is not 0), the MD5 algorithm cannot be used to create the fingerprints. The MD5 entry shows 15 colons like this example, "::::::::::::::::".

____

3. Optionally insert a comment.

____

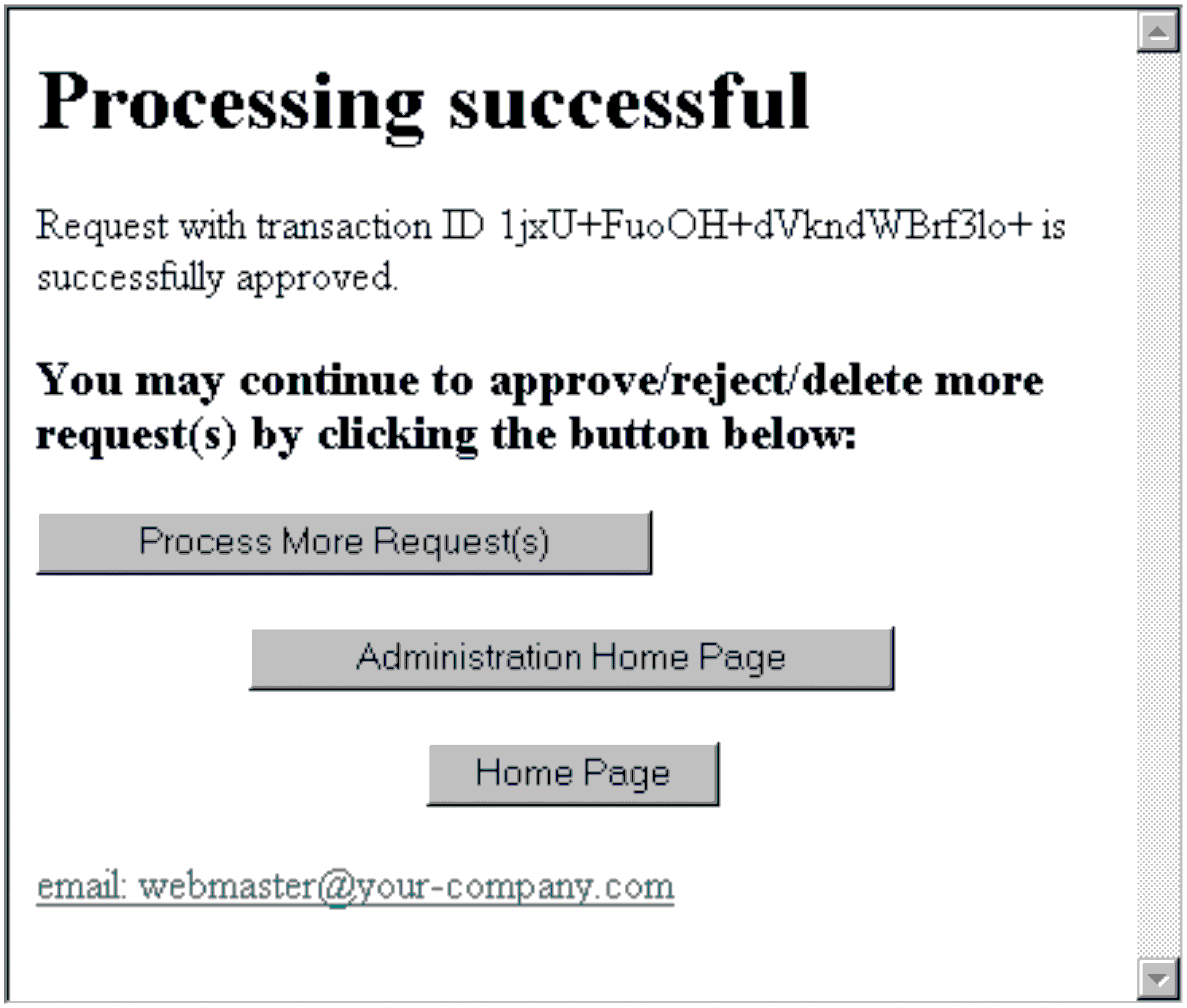
4. Click one of the choices on the "Single Request" web page to process the request.

   - **Approve the request as is**
   - **Approve the request with modifications**
   - **Reject request**
   - **Delete request**

   The choices on the "Single Request" web page appear based on the status of the request. For example:

   - If the status of a request is `Pending Approval`, only the first three choices in the preceding list appear.
   - If the administrator has already processed the request or if the request is `Preregistered`, only **Delete request** appears.

   a. When you click **Approve the request as is** and processing is successful, the result is a web page that says "Processing is successful", such as the one shown in Figure 70 on page 414. (Otherwise, the web page says "Processing is not successful".)



Processing successful

Request with transaction ID 1jxU+FuoOH+dVkndWBrf3lo+ is successfully approved.

You may continue to approve/reject/delete more request(s) by clicking the button below:

Figure 70. Processing successful web page

From these web pages, you can then click **Process more request(s)** to return to the PKI Services administration home page (Figure 68 on page 411).

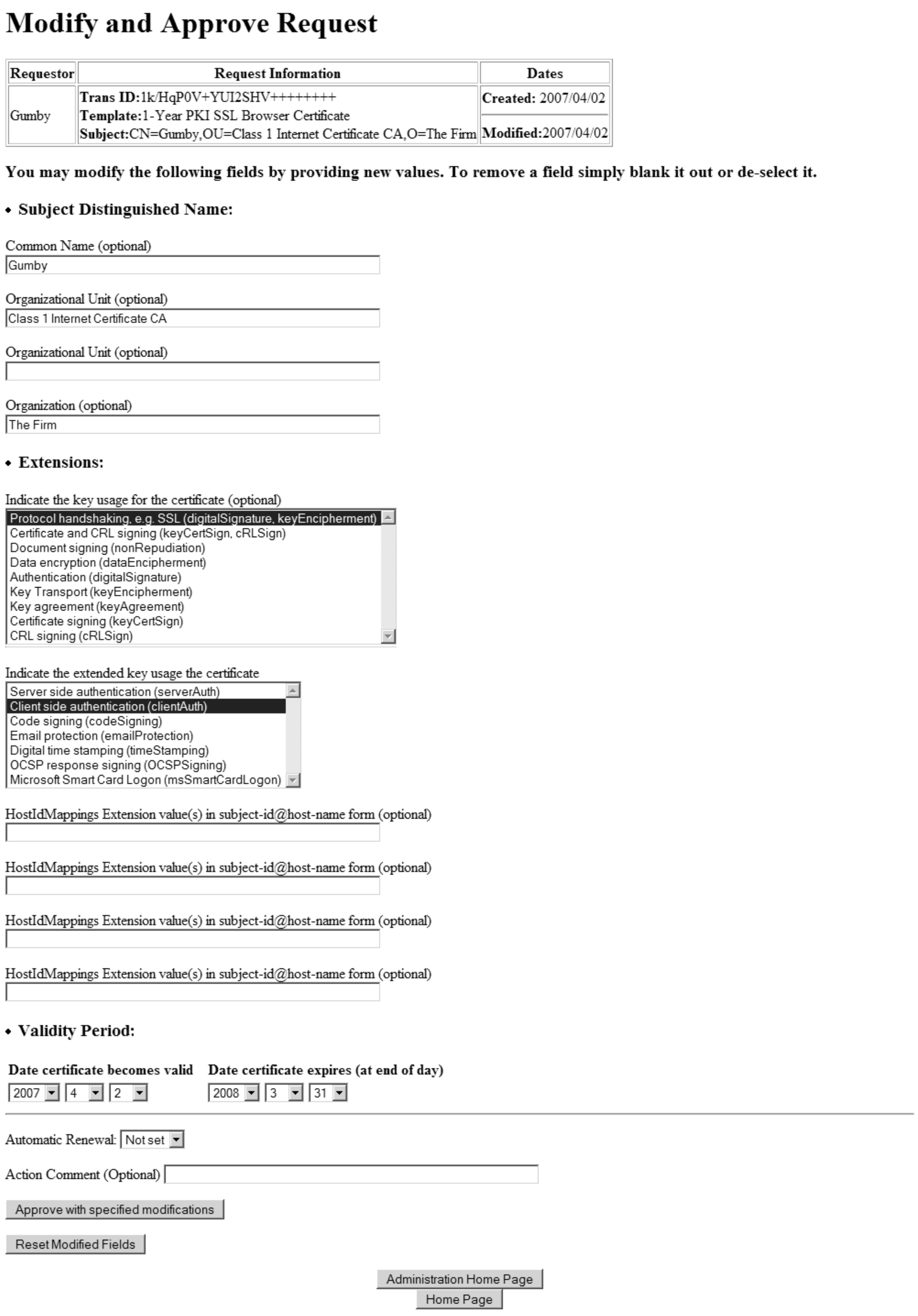
b. When you click **Approve the request with modifications**, this displays a "Modify and Approve Request" web page similar to the one shown in Figure 72 on page 415.

If the subject's distinguished name contained in the current request is not in the proper format for RACF processing, you see the note (Figure 71 on page 414) on the "Modify and Approve Request" web page.

Note - the existing subject's name is not in a format that can be re-created by PKI Services. Therefore, specifying any subject's name field below causes the existing name to be deleted and completely replaced.

Figure 71. Restriction note on the modify and approve request web page

Restriction: If you receive the note that is shown in Figure 71 on page 414, you cannot change any field of the subject's distinguished name (the common name, organizational unit, or organization field) without causing PKI Services to delete the entire subject's distinguished name and replace it with your changed values. (This is because the subject's distinguished name is not in the proper format for RACF processing.)


Modify and Approve Request

Requestor	Request Information	Dates
Gumby	**Trans ID:**1k/HqP0V+YUI2SHV++++++++ **Template:**1-Year PKI SSL Browser Certificate **Subject:**CN=Gumby,OU=Class 1 Internet Certificate CA,O=The Firm	**Created:** 2007/04/02 **Modified:**2007/04/02

You may modify the following fields by providing new values. To remove a field simply blank it out or de-select it.

- **Subject Distinguished Name:**

Common Name (optional): Gumby

Organizational Unit (optional): Class 1 Internet Certificate CA

Organizational Unit (optional):

Organization (optional): The Firm

- **Extensions:**

Indicate the key usage for the certificate (optional)

- Protocol handshaking, e.g. SSL (digitalSignature, keyEncipherment) (selected)
- Certificate and CRL signing (keyCertSign, cRLSign)
- Document signing (nonRepudiation)
- Data encryption (dataEncipherment)
- Authentication (digitalSignature)
- Key Transport (keyEncipherment)
- Key agreement (keyAgreement)
- Certificate signing (keyCertSign)
- CRL signing (cRLSign)

Indicate the extended key usage the certificate

- Server side authentication (serverAuth)
- Client side authentication (clientAuth) (selected)
- Code signing (codeSigning)
- Email protection (emailProtection)
- Digital time stamping (timeStamping)
- OCSP response signing (OCSPSigning)
- Microsoft Smart Card Logon (msSmartCardLogon)

HostIdMappings Extension value(s) in subject-id@host-name form (optional):

HostIdMappings Extension value(s) in subject-id@host-name form (optional):

HostIdMappings Extension value(s) in subject-id@host-name form (optional):

HostIdMappings Extension value(s) in subject-id@host-name form (optional):

- **Validity Period:**

Date certificate becomes valid: 2007 4 2 **Date certificate expires (at end of day)**: 2008 3 31

Automatic Renewal: Not set

Action Comment (Optional):

Figure 72. Modifying the request web page

On the "Modify and Approve Request" web page, you can change the following fields.

- Common name, unless noted as in Figure 71 on page 414
- Organizational unit (this can be multiple fields), unless noted as in Figure 71 on page 414
- Organization, unless noted as in Figure 71 on page 414
- Email address

  Note: If you change the value of the email address field (email) and if the original request included the notification email address field (NotifyEmail), the value of the latter field is changed to match the changed email address value.
- Street
- Postal code
- Certificate purpose
- Date certificate becomes valid
- Date certificate expires
- `HostIdMappings` extensions (This can be multiple fields.)
- Optional comment about action you perform on the certificate.
- Automatic renewal

When you are satisfied with the changes you have made, click **Approve with specified modifications**; or, if you change your mind, you can click **Reset modified fields**. Alternately, you can click **Home page** to go to the PKI Services home page. (See "#unique_66/unique_66_Connect_42_mainpage" on page 376.)

Note: When you click Approve with specified modifications for a request that requires multiple approvals, any previous made approvals are nullified. The request remains in Pending Approval state, and the current approval count for the request is set to 1.

c. When you click **Reject request**, this displays a web page that informs you that "Processing is successful" or that "Processing is not successful". From these web pages, click **Process more request(s)** to return to the PKI Services administration home page. (See Figure 68 on page 411.)

d. When you click **Delete request**, this displays a web page that informs you that "Processing is successful" or that "Processing is not successful". On these web pages, click **Process more request(s)** to return to the PKI Services administration home page. (See Figure 68 on page 411.)

____

Steps for processing requests by performing searches

The administrator can use the web page to search for certificate requests of various statuses. The following table summarizes the searches that are listed on the web page and the certificate requests that are displayed as a result:

Table 71. Searches to display certificate requests

Search criteria	Results
Show all requests	Displays all certificate requests (all statuses (`Pending Approval`, `Approved`, `Completed`, `Preregistered`, `Rejected`, or `Rejected, User Notified`).
Show requests pending approval	Displays only certificate requests whose status is `Pending Approval`.
Show approved requests	Displays certificate requests whose status is `Approved` or `Completed`.
Show completed requests	Displays certificate requests whose status is `Completed`.
Show preregistered requests	Display certificate requests whose status is `Preregistered`.

Table 71. Searches to display certificate requests (continued)

Search criteria	Results
Show all rejected requests	Displays certificate requests whose status is `Rejected`, or `Rejected, User Notified`.
Show rejections in which the client has been notified	Displays certificate requests whose status is `Rejected, User Notified`.

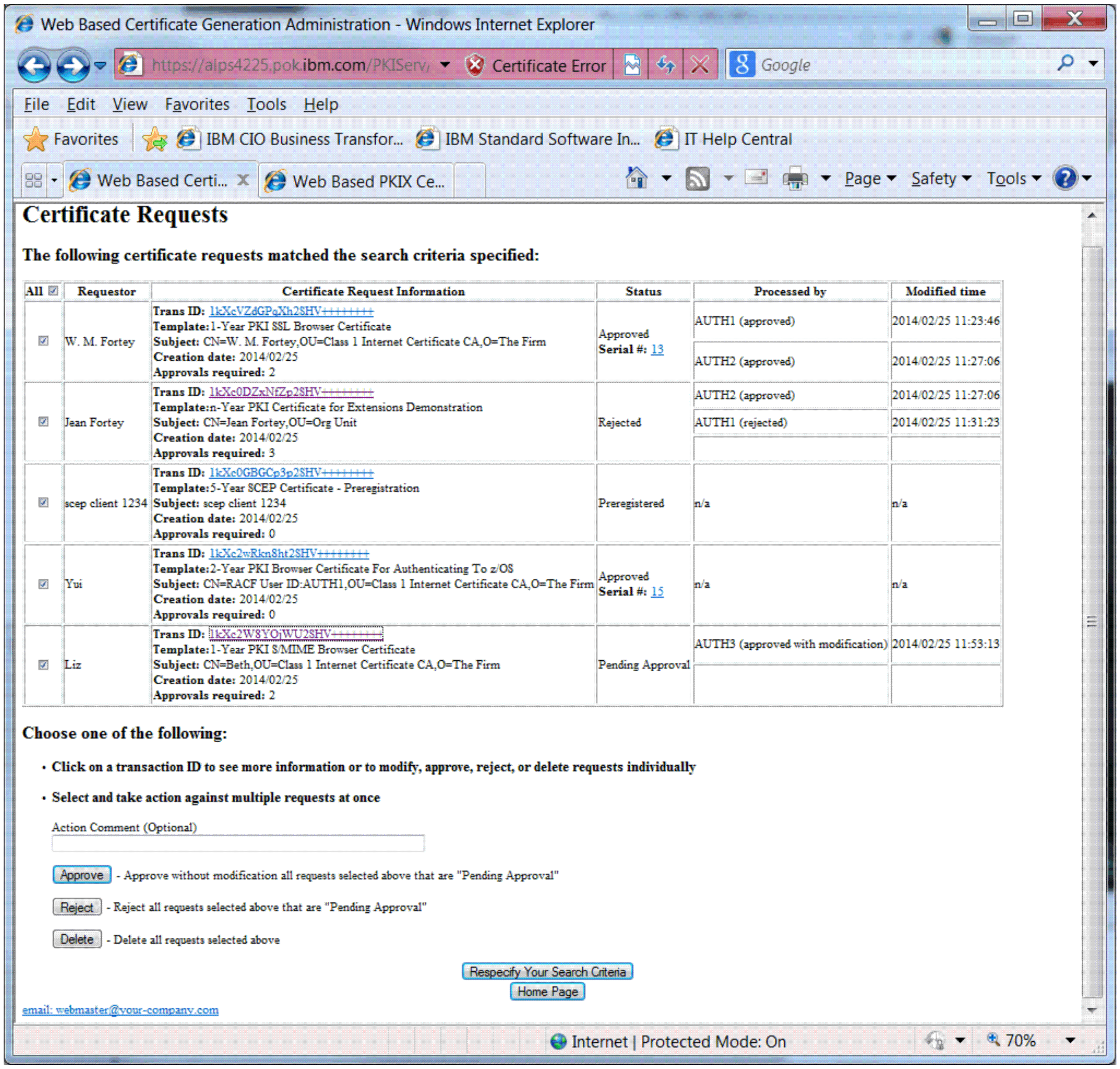
To process requests by performing a search for requests of a particular status, perform the following steps:

1. On the PKI Services administration home page (see Figure 68 on page 411), select one of the searches by clicking the appropriate choice under **Certificate Requests**. (Table 71 on page 416 describes these searches.) You can optionally fill in additional search criteria (**Requestor's name** and **Show recent activity only**).

   Guideline: Queries against the request database might time out if the database contains a large number of records. The performance of the query can be vastly improved by supplying **Requestor's name** as additional search criteria if the saved requestor data is meaningful to your organization and it is recallable. In this case, a PKI exit can be used to supply a meaningful value, such as a Lotus Notes short name or customer account number.

   ____

2. Click **Find certificates or certificate requests**. The following window opens:



Certificate Requests

The following certificate requests matched the search criteria specified:

All ☑	Requestor	Certificate Request Information	Status	Processed by	Modified time
☑	W. M. Fortey	**Trans ID:** 1kXcVZdGPqXh2SHV++++++++ **Template:**1-Year PKI SSL Browser Certificate **Subject:** CN=W. M. Fortey,OU=Class 1 Internet Certificate CA,O=The Firm **Creation date:** 2014/02/25 **Approvals required:** 2	Approved **Serial #:** 13	AUTH1 (approved) AUTH2 (approved)	2014/02/25 11:23:46 2014/02/25 11:27:06
☑	Jean Fortey	**Trans ID:** 1kXc0DZxNfZp2SHV++++++++ **Template:**n-Year PKI Certificate for Extensions Demonstration **Subject:** CN=Jean Fortey,OU=Org Unit **Creation date:** 2014/02/25 **Approvals required:** 3	Rejected	AUTH2 (approved) AUTH1 (rejected)	2014/02/25 11:27:06 2014/02/25 11:31:23
☑	scep client 1234	**Trans ID:** 1kXc0GBGCp3p2SHV++++++++ **Template:**5-Year SCEP Certificate - Preregistration **Subject:** scep client 1234 **Creation date:** 2014/02/25 **Approvals required:** 0	Preregistered	n/a	n/a
☑	Yui	**Trans ID:** 1kXc2wRkn8ht2SHV++++++++ **Template:**2-Year PKI Browser Certificate For Authenticating To z/OS **Subject:** CN=RACF User ID:AUTH1,OU=Class 1 Internet Certificate CA,O=The Firm **Creation date:** 2014/02/25 **Approvals required:** 0	Approved **Serial #:** 15	n/a	n/a
☑	Liz	**Trans ID:** 1kXc2W8YOjWU2SHV++++++++ **Template:**1-Year PKI S/MIME Browser Certificate **Subject:** CN=Beth,OU=Class 1 Internet Certificate CA,O=The Firm **Creation date:** 2014/02/25 **Approvals required:** 2	Pending Approval	AUTH3 (approved with modification)	2014/02/25 11:53:13

Choose one of the following:

- **Click on a transaction ID to see more information or to modify, approve, reject, or delete requests individually**
- **Select and take action against multiple requests at once**

Action Comment (Optional)

Approve - Approve without modification all requests selected above that are "Pending Approval"

Reject - Reject all requests selected above that are "Pending Approval"

Delete - Delete all requests selected above

Respecify Your Search Criteria

Home Page

email: webmaster@your-company.com

Figure 73. Processing requests after searching

Note: The table at the beginning of the web page shows the certificate requests that match your search criteria. (If multiple certificates requests match the search criteria, up to ten appear on a web page, and a button at the bottom of the web age allows you to view the next set.)

____

3. You can use this web page:
   - To process a single certificate request
   - To perform the same action on all of the certificate requests that are listed
   - To process selected requests.

   To process a single certificate request:

   a. Click on its transaction ID in the table at the beginning of the web page. This transfers you to the **Single Request** web page; see Figure 69 on page 412.
   b. From the **Single Request** web page, you can perform the steps in the preceding section, starting with Step “2” on page 412).

   To perform the same action on all the certificate requests that are listed:

a. Optionally enter a comment.

b. Click one of the actions under in the comment field to perform that action on all listed requests:

   Approve
   Approves without modification all requests that are pending approval.

   Reject
   Rejects all requests that are pending approval.

   Delete
   Deletes all requests.

   Note: The **Approve** and **Reject** actions appear only if certificate requests are pending approval. Otherwise, only the **Delete** action appears.

To process selected certificate requests:

a. Clear the check box beside the **Select** column header. (When the check box beside **Select** is selected, all the individual check boxes in the body of the table are selected. This means all these certificate requests are selected. Clearing the box in the header clears all the boxes in the body of the table.)

b. Select the check boxes of all the certificate requests for which you want to perform a particular action.

c. Optionally enter a comment.

d. Click one of the actions under the comment field to perform that action on all listed requests. The actions include:

   Approve
   Approves without modification all requests that are pending approval.

   Reject
   Rejects all requests that are pending approval.

   Delete
   Deletes all requests.

   Note: The **Approve** and **Reject** actions appear only if certificate requests are pending approval. Otherwise, only the **Delete** action appears.

Tip: If you select **Show all requests** (see Figure 68 on page 411) and click **Approve** on this web page, only the certificate requests whose status is `Pending Approval` are approved.

Instead of processing one or more certificate requests, you can click **Respecify your search criteria web page** to return to the PKI Services administration home page (see Figure 68 on page 411) or **Home page** to return to the PKI Services home page (see "#unique_66/unique_66_Connect_42_mainpage" on page 376).

____

4. After you click an action, you see one of the following web pages;

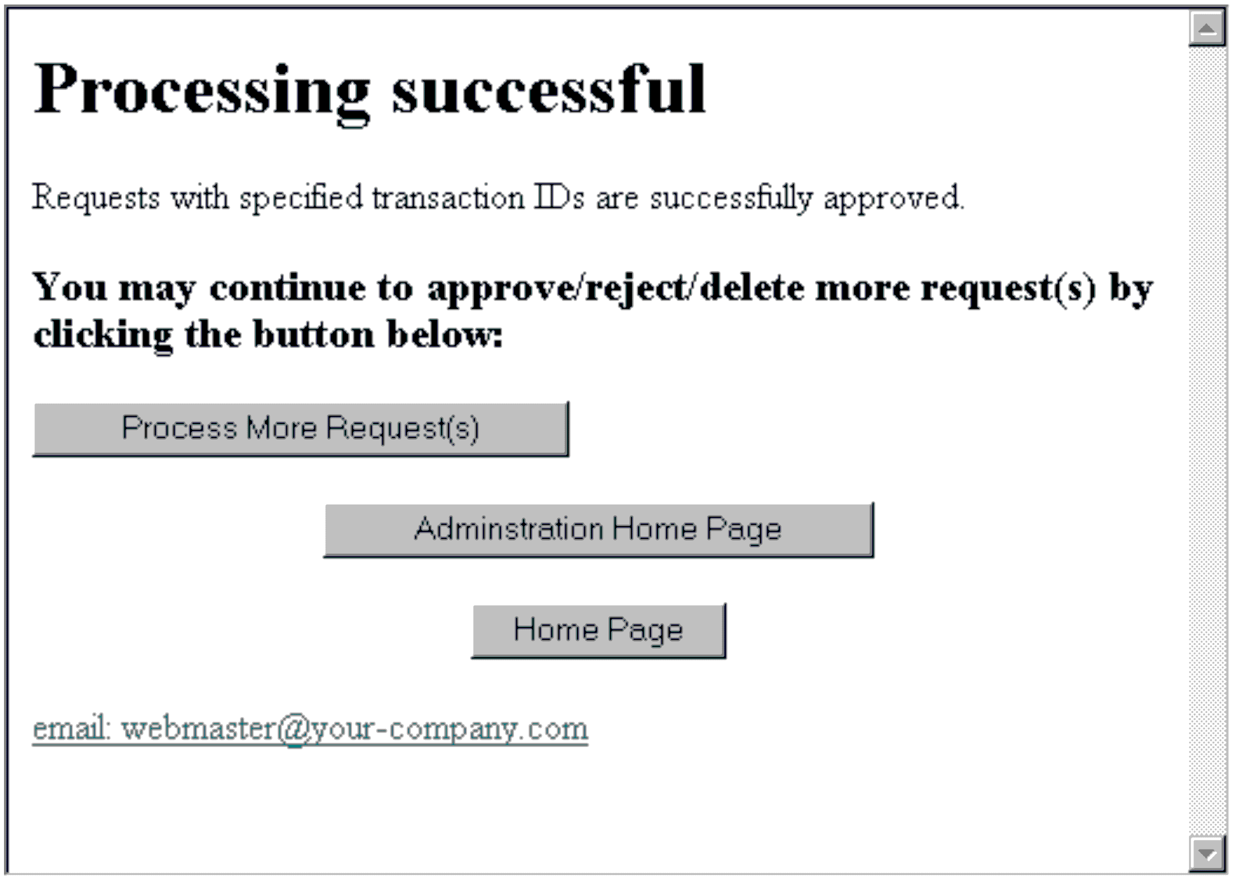
   - **Processing successful** (see Figure 74 on page 420)
   - **Processing was not successful** (see Figure 75 on page 421)
   - **Processing partially successful** (see Figure 76 on page 421)

   If **Processing was not successful**, you can click on the transaction ID to display the **Single Request** web page; see Figure 69 on page 412. Processing can be unsuccessful because requests do not have the status required for the action you selected; see Table 70 on page 410.

   If you get **Processing partially successful**, you can click on the transaction ID to display the **Single Request** web page; see Figure 69 on page 412. This message can occur when your organization has more than one administrator and involves the following sequence:

   a. One administrator performs a search.

b. Another administrator performs a search before the first administrator has approved requests displayed in the search results.
c. One of the administrators approves only some of the requests.
d. The other administrator tries to approve requests including at least one the preceding administrator has already approved and one that the preceding administrator has not already approved.

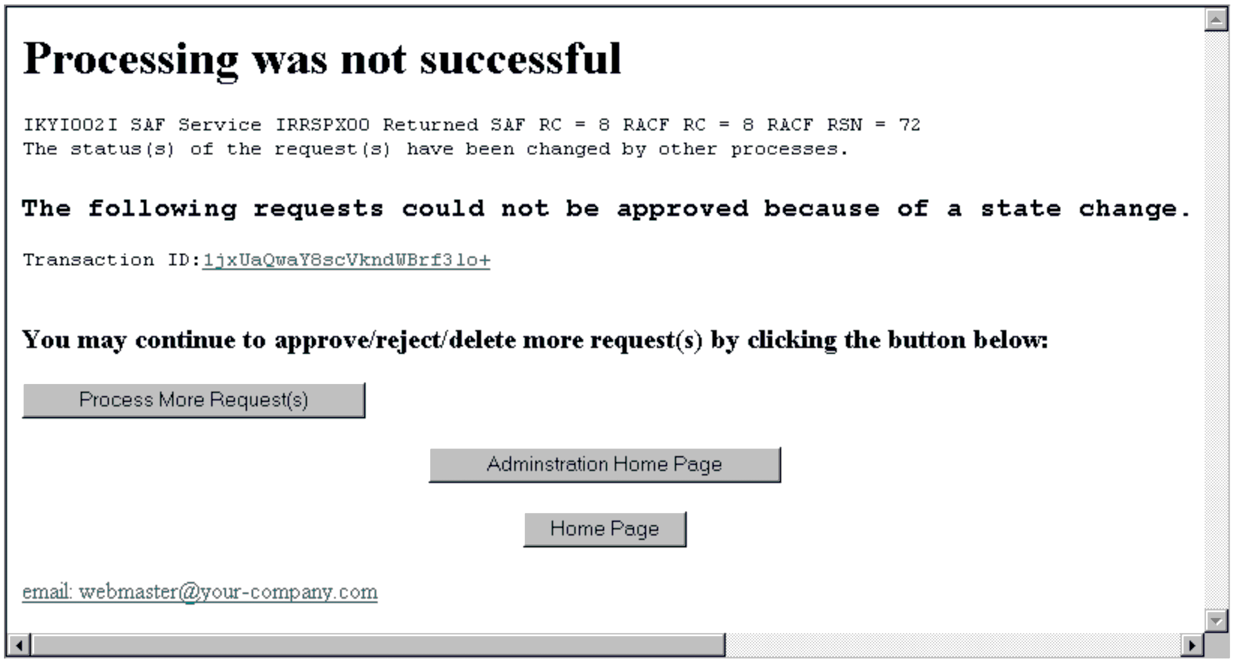


Processing successful

Requests with specified transaction IDs are successfully approved.

You may continue to approve/reject/delete more request(s) by clicking the button below:

Figure 74. Request processing was successful web page



Processing was not successful

```
IKYI002I SAF Service IRRSPX00 Returned SAF RC = 8 RACF RC = 8 RACF RSN = 72
The status(s) of the request(s) have been changed by other processes.
```

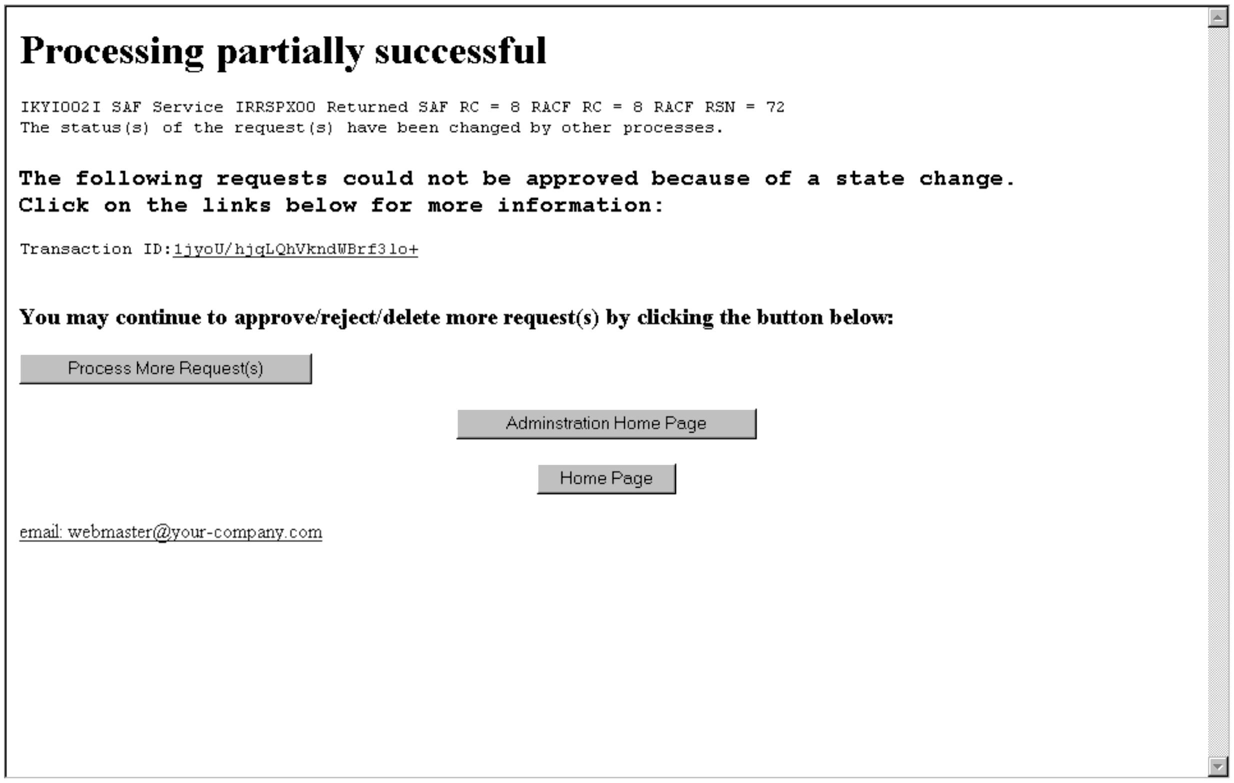
The following requests could not be approved because of a state change.

Transaction ID:1jxUaQwaY8scVkndWBrf3lo+

You may continue to approve/reject/delete more request(s) by clicking the button below:

email: webmaster@your-company.com

Figure 75. Request processing was not successful web page



Processing partially successful

```
IKYI002I SAF Service IRRSPX00 Returned SAF RC = 8 RACF RC = 8 RACF RSN = 72
The status(s) of the request(s) have been changed by other processes.
```

The following requests could not be approved because of a state change.
Click on the links below for more information:

Transaction ID:1jyoU/hjqLQhVkndWBrf3lo+

You may continue to approve/reject/delete more request(s) by clicking the button below:

email: webmaster@your-company.com

Figure 76. Request processing was partially successful web page

5. After approving requests as appropriate, you can:
   - Click **Process more request(s)** to return to Figure 73 on page 418.
   - Click **Administration home page** to return to Figure 68 on page 411.
   - Click **Home page** to return to “#unique_66/unique_66_Connect_42_mainpage” on page 376.

Processing certificates

Before you can use the web page to process certificates, you need to understand the statuses of certificates and actions you can perform on certificates.

Status of certificates

Certificates that have been created from requests are maintained permanently in an issued certificate database. Another name for this is the issued certificate list (ICL). Issued certificates are also published in an LDAP directory.

A certificate can have only one of the following states (statuses) at a time:

Table 72. Status of certificates

Status	**Meaning**
`Active`	The certificate has not yet expired, has not been revoked, and is not currently suspended.
`Active, AutoRenew`	The certificate is eligible for automatic renewal and this capability is enabled.
`Active, AutoRenewDisabled`	The certificate is eligible for automatic renewal but this capability is disabled.
`Active, NotRenewable`	The certificate cannot be renewed.
`Expired`	The certificate's validity period expired while it was active.
`Revoked`	The certificate has not expired but it has been revoked. Such certificates are published on the next certificate revocation list (CRL).
`Revoked, Expired`	The certificate was either revoked or suspended, and time has elapsed so that it is now also expired. Such certificates are not published on the next CRL.
`Suspended`	The certificate has not expired but it is currently suspended. Such certificates are published on the next certificate revocation list (CRL).

The administrator must approve a request for the certificate to have a status (as enumerated in the preceding list) or for the administrator to delete the certificate from the ICL. (An administrator can delete a certificate from the ICL, but this would not be a normal situation.) Alternately, the administrator can reject a request or delete the request from the request database (RDB). If the administrator does not approve the request, it is never listed in the ICL.

Actions for certificates

The following table summarizes actions on certificates and the required status to perform these actions:

Table 73. Summary of actions to perform and required status to do so

Action	**Required status of certificate**	**Who performs action**
Renew	`Active`	End user
Resume	`Suspended`	Administrator
Revoke	`Active` or `Suspended`	End user or administrator
Suspend	`Active`	End user or administrator

Table 73. Summary of actions to perform and required status to do so (continued)

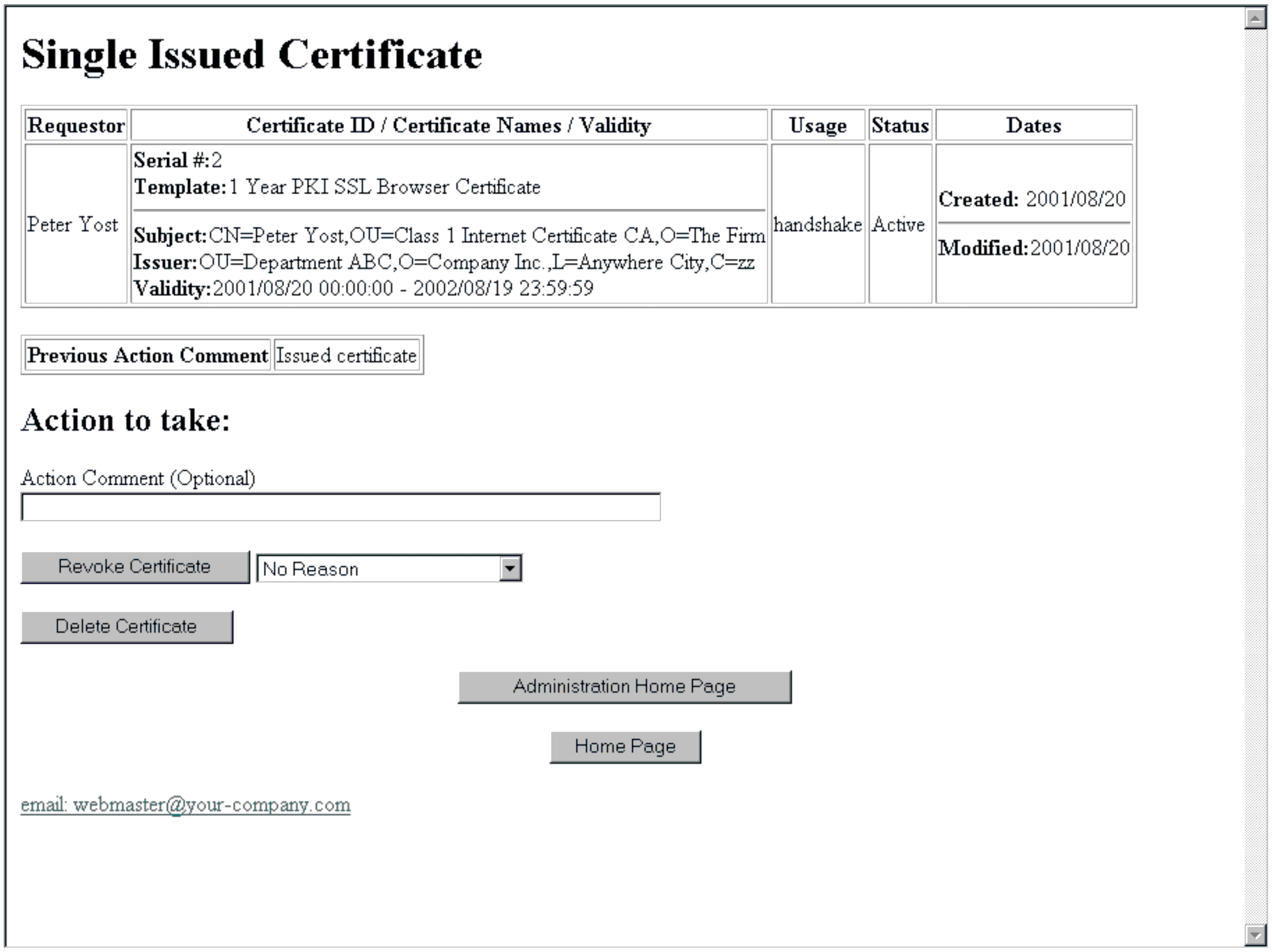
Action	**Required status of certificate**	**Who performs action**
Delete	All statuses (`Active`, `Expired`, `Suspended`, `Revoked`, or `Revoked, Expired`)	Administrator
Enable automatic renewal	`Active or Active,AutoRenewDisabled`	Administrator
Disable automatic renewal	`Active or Active,AutoRenew`	Administrator
Change requester email	All statuses **Note:** This action applies only to certificates for which PKI Services generated the key pair.	Administrator

Note: You can resume (reactivate) a suspended certificate, or permanently revoke it, if the certificate has not yet expired and the suspension grace period has not elapsed. If the grace period has elapsed, the certificate is permanently revoked the next time certificate revocation lists (CRLs) are issued.

Steps for processing a single certificate

To process a single certificate, perform the following steps:

1. On the PKI Services administration home page (see Figure 68 on page 411), enter the serial number of the certificate you want to process in the field provided for it. Figure 77 on page 424 shows an example of the web page that is displayed:



Single Issued Certificate

Requestor	Certificate ID / Certificate Names / Validity	Usage	Status	Dates
Peter Yost	**Serial #:**2 **Template:**1 Year PKI SSL Browser Certificate **Subject:**CN=Peter Yost,OU=Class 1 Internet Certificate CA,O=The Firm **Issuer:**OU=Department ABC,O=Company Inc.,L=Anywhere City,C=zz **Validity:**2001/08/20 00:00:00 - 2002/08/19 23:59:59	handshake	Active	**Created:** 2001/08/20 **Modified:**2001/08/20

Previous Action Comment	Issued certificate

Action to take:

Action Comment (Optional)

email: webmaster@your-company.com

Figure 77. Processing a certificate from the single certificate web page

2. Make sure the certificate is the correct one by reviewing the information in the upper part of the web page.

3. If you are going to process a certificate from this web page, you can optionally insert a comment.

4. Click one of the following actions to process the certificate:

Disable automatic renewal
Disables automatic renewal for the certificate

Enable automatic renewal
Enables automatic renewal for the certificate

Revoke certificate
Revokes the certificate.

Suspend certificate
Suspends the certificate.

Resume certificate
Resumes the certificate.

Delete certificate
Deletes the certificate. (This is for cleanup purposes.)

Change Requestor email
Changes the requestor's email to the value that you supply in the field next to the button. Use this function when the email address of the requestor of a PKI generated key certificate has changed. If you change the requestor's email, the certificate cannot be renewed.

Note:

a. The **Suspend** and **Revoke** buttons appear only if the status of the certificate is `Active`.
b. The **Resume** button appears only if the status of the certificate is `Suspended`.
c. The **Enable Automatic Renewal** button appears only if the status of the certificate is `Active,AutoRenwDisabled`.
d. The **Disable Automatic Renewal** button appears only if the status of the certificate is `Active,AutoRenew`.
e. The **Change Requestor email** button appears only for a PKI generated key certificate.

____

Steps for processing certificates by performing searches

The administrator can use the web page to search for certificates of various statuses. The following table summarizes the searches listed on the web page and the certificates that are displayed as a result:

Table 74. Searches to display certificates

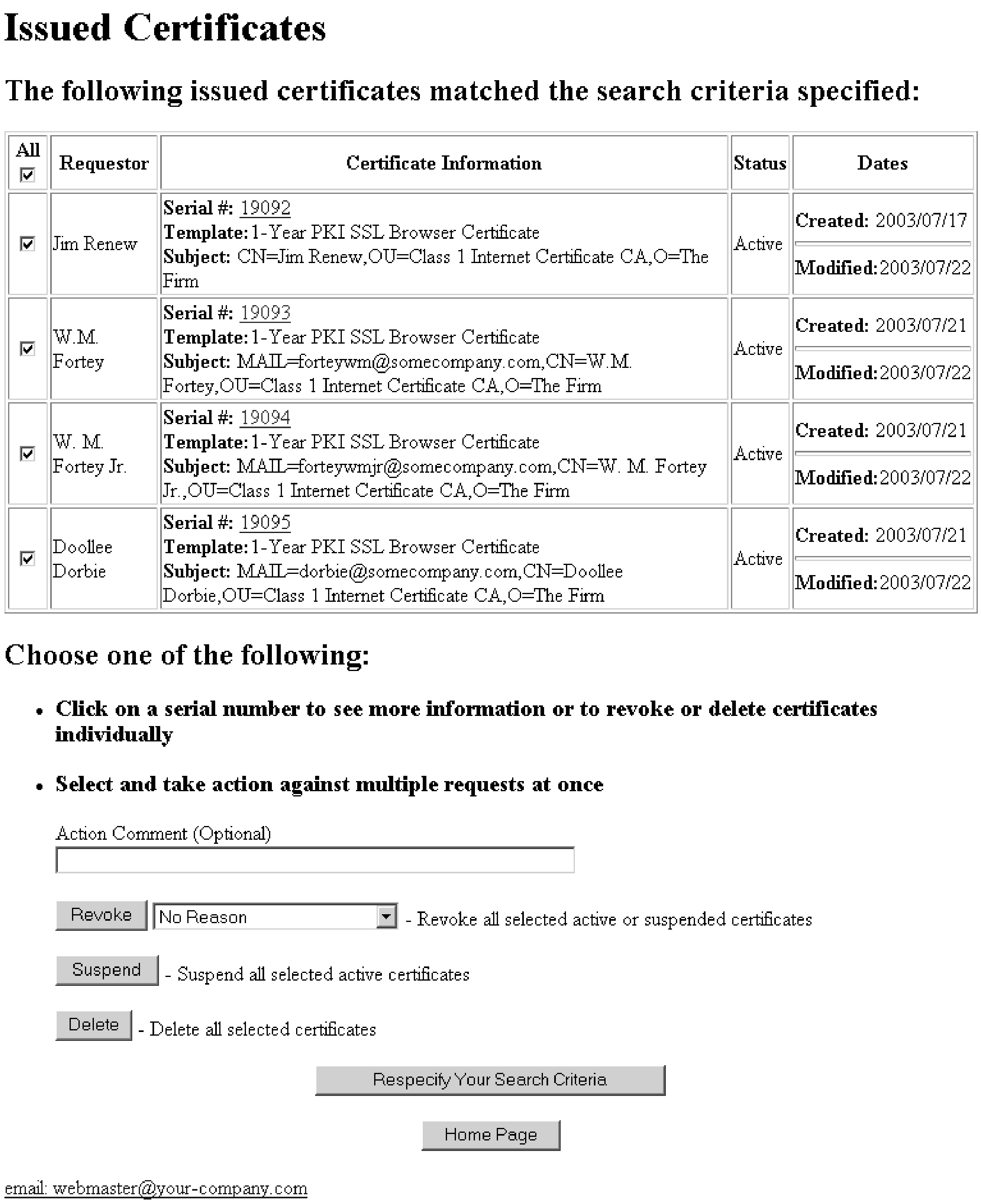
Searches	Results
Show all issued certificates.	Displays all certificates (can be any status).
Show revoked certificates.	Displays certificates whose status is `Revoked` or `Revoked, Expired`.
Show suspended certificates.	Displays certificates whose status is `Suspended`.
Show expired certificates.	Displays certificates whose status is `Expired` or `Revoked, Expired`.
Show active certificates (not expired, not revoked, not suspended).	Displays certificates whose status is `Active`.
Show disabled certificates (suspended or revoked, not expired).	Displays certificate requests whose status is `Suspended` or `Revoked`.
Show certificates with automatic renewal enabled.	Displays certificates whose status is `Active,AutoRenew`.
Show certificates with automatic renewal disabled.	Displays certificates whose status is `Active,AutoRenewDisabled`.
Show certificates that cannot be renewed.	Displays certificates whose status is `Active,NotRenewable`.

To process certificates by performing a search for certificates of a particular status, perform the following steps:

1. On the PKI Services administration home page (see Figure 68 on page 411), select one of the searches by clicking the appropriate choice under **Issued Certificates**. (The preceding table describes these searches.) You can optionally fill in additional search criteria (**Requestor's name** and **Show recent activity only**).

   ____

2. Click **Find certificates or certificate requests**. This displays the following web page.



Issued Certificates

The following issued certificates matched the search criteria specified:

All ✓	Requestor	Certificate Information	Status	Dates
✓	Jim Renew	**Serial #:** 19092 **Template:** 1-Year PKI SSL Browser Certificate **Subject:** CN=Jim Renew,OU=Class 1 Internet Certificate CA,O=The Firm	Active	**Created:** 2003/07/17 **Modified:** 2003/07/22
✓	W.M. Fortey	**Serial #:** 19093 **Template:** 1-Year PKI SSL Browser Certificate **Subject:** MAIL=forteywm@somecompany.com,CN=W.M. Fortey,OU=Class 1 Internet Certificate CA,O=The Firm	Active	**Created:** 2003/07/21 **Modified:** 2003/07/22
✓	W. M. Fortey Jr.	**Serial #:** 19094 **Template:** 1-Year PKI SSL Browser Certificate **Subject:** MAIL=forteywmjr@somecompany.com,CN=W. M. Fortey Jr.,OU=Class 1 Internet Certificate CA,O=The Firm	Active	**Created:** 2003/07/21 **Modified:** 2003/07/22
✓	Doollee Dorbie	**Serial #:** 19095 **Template:** 1-Year PKI SSL Browser Certificate **Subject:** MAIL=dorbie@somecompany.com,CN=Doollee Dorbie,OU=Class 1 Internet Certificate CA,O=The Firm	Active	**Created:** 2003/07/21 **Modified:** 2003/07/22

Choose one of the following:

- **Click on a serial number to see more information or to revoke or delete certificates individually**
- **Select and take action against multiple requests at once**

Action Comment (Optional)

- Revoke all selected active or suspended certificates

- Suspend all selected active certificates

- Delete all selected certificates

email: webmaster@your-company.com

Figure 78. Processing certificates using searches

Note: The table at the beginning of the web page shows the certificates that match your search criteria. (If multiple certificates match the search criteria, up to ten appear on a web page, and a button at the bottom of the web page allows you to view the next set.)

3. You can use this web page:
   - To process a single certificate
   - To perform the same action on all of the certificates that are listed

- To process selected certificate

To process a single certificate:

a. Click on its serial number in the table at the beginning of the web page. This transfers you to the single certificate web page; see Figure 77 on page 424.
b. From the single certificate web page, you can perform the steps in "Steps for processing a single certificate" on page 423, starting with Step "2" on page 424).

To perform the same action on all the certificates that are listed:

a. Optionally enter a comment.
b. Click one of the actions under the comment field to perform that action on all listed certificates:

Revoke
Revokes all selected active certificates.

Suspend
Suspends all selected active certificates.

Resume
Resumes all selected suspended certificates.

Delete
Deletes all selected certificates.

Notes:

1) **Suspend** and **Revoke** appear only when your search matches at least one certificate whose status is `Active`.
2) **Resume** appears only when your search matches at least one certificate whose status is `Suspended`.
3) **Enable Automatic Renewal**, **Disable Automatic Renewal**, and **Change Requestor email** are not shown on this page. They only appear on the page showing the individual certificate.

To process selected certificates:

a. Clear the check box beside the **Select** column header. (When the check box beside **Select** is selected, all the individual check boxes in the body of the table are selected. This means all these certificates are selected. Clearing the box in the header clears all the boxes in the body of the table.)
b. Select the check boxes of all the certificates for which you want to perform a particular action.
c. Optionally enter a comment.
d. Click one of the actions under the comment field to perform that action on all listed requests. The actions include:

Revoke
Revokes all selected active certificates.

Suspend
Suspends all selected active certificates.

Resume
Resumes all selected suspended certificates.

Delete
Deletes all selected certificates.

Notes:

1) The **Suspend** and **Revoke** actions appear only when your search matches at least one certificate whose status is `Active`.
2) The **Resume** action appears only when your search matches at least one certificate whose status is `Suspended`.

Instead of processing one or more certificates, you can click **Respecify your search criteria Web page** to return to the PKI Services administration home page (see Figure 68 on page 411) or **Home page** to return to the PKI Services home page (see "#unique_66/unique_66_Connect_42_mainpage" on page 376.)

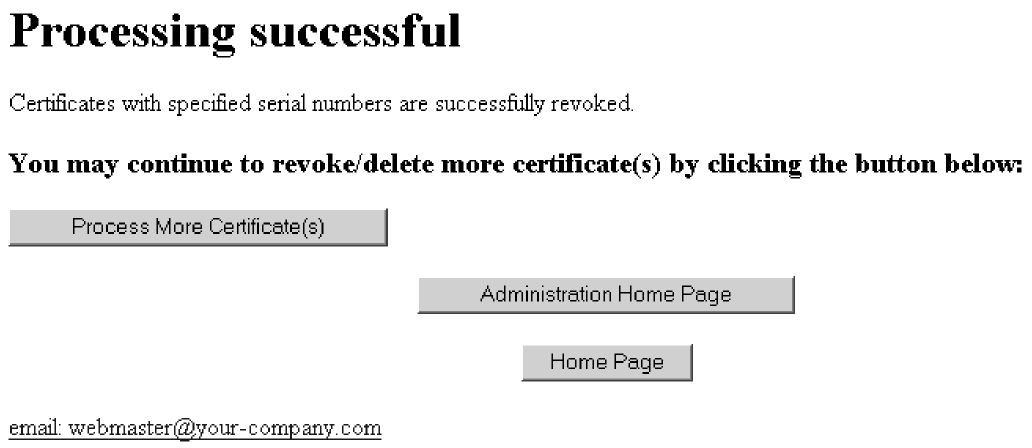
____

4. After you click an action, the next web page tells you:
   - "Processing was successful" (see Figure 79 on page 428)
   - "Processing was not successful" (see Figure 80 on page 429)
   - "Processing was partially successful" (see Figure 81 on page 429)

   If "Processing was not successful", you can click on a serial number to display the **Single Certificate** web page; see Figure 77 on page 424. Processing can be unsuccessful because certificates do not have the status required for the action you selected; see Table 73 on page 422.

   If you get "Processing partially successful", you can click on the serial number to display the **Single Certificate** web page; see Figure 77 on page 424. The "Processing partially successful" message can occur when your organization has more than one administrator and involves the following sequence:

   a. One administrator performs a search.
   b. Another administrator performs a search before the first administrator has revoked or deleted certificates displayed in the search results.
   c. One of the administrators revokes or deletes some of the certificates.
   d. The other administrator tries to revoke or delete certificates including at least one the preceding administrator has already revoked or deleted and at least one the preceding administrator has not already revoked or deleted.

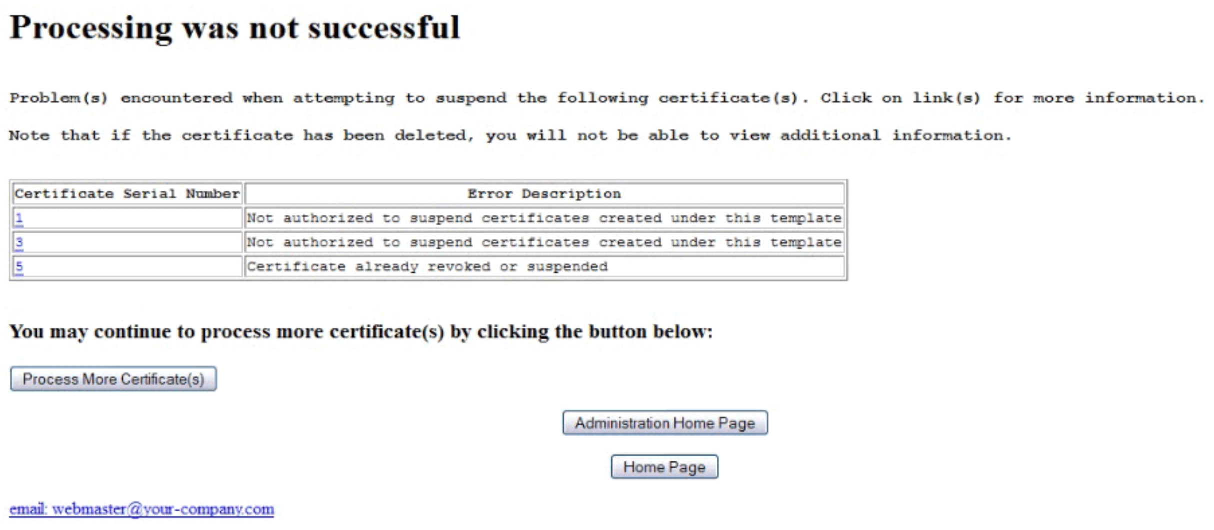

Processing successful

Certificates with specified serial numbers are successfully revoked.

You may continue to revoke/delete more certificate(s) by clicking the button below:

email: webmaster@your-company.com

Figure 79. Processing of certificate was successful web page



Processing was not successful

Problem(s) encountered when attempting to suspend the following certificate(s). Click on link(s) for more information.

Note that if the certificate has been deleted, you will not be able to view additional information.

Certificate Serial Number	Error Description
1	Not authorized to suspend certificates created under this template
3	Not authorized to suspend certificates created under this template
5	Certificate already revoked or suspended

You may continue to process more certificate(s) by clicking the button below:

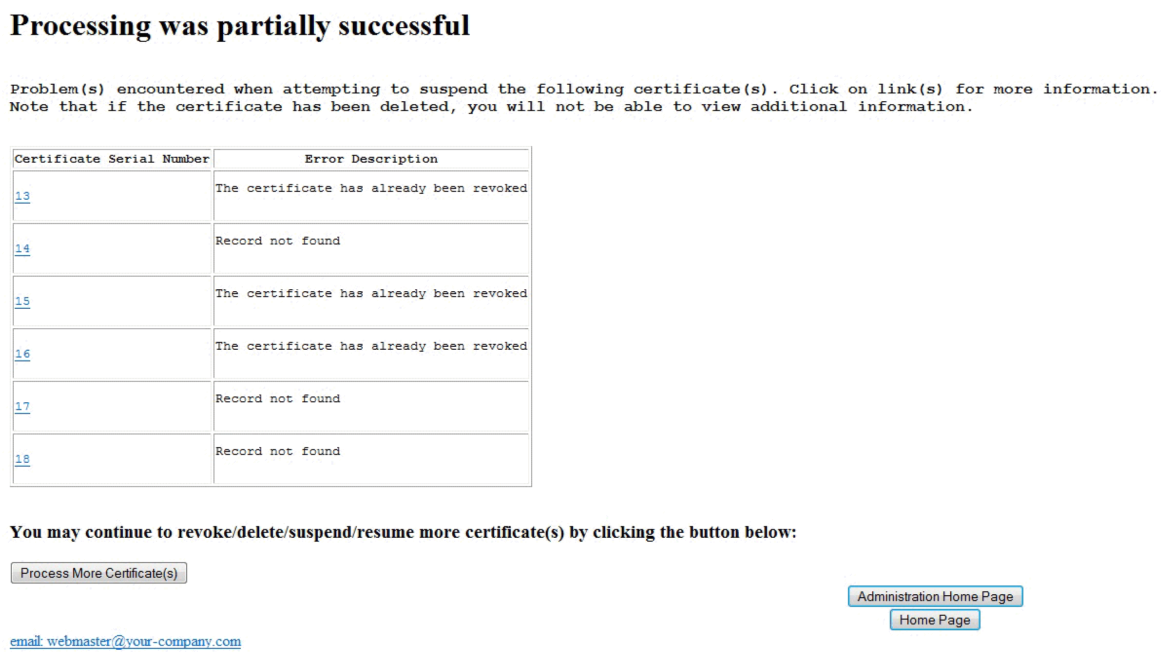
Process More Certificate(s)

Administration Home Page

Home Page

email: webmaster@your-company.com

Figure 80. Processing of certificate was not successful web page



Processing was partially successful

Problem(s) encountered when attempting to suspend the following certificate(s). Click on link(s) for more information.
Note that if the certificate has been deleted, you will not be able to view additional information.

Certificate Serial Number	Error Description
13	The certificate has already been revoked
14	Record not found
15	The certificate has already been revoked
16	The certificate has already been revoked
17	Record not found
18	Record not found

You may continue to revoke/delete/suspend/resume more certificate(s) by clicking the button below:

Process More Certificate(s)

Administration Home Page

Home Page

email: webmaster@your-company.com

Figure 81. Processing of certificate was partially successful web page

You can click **Home page** to return you to the PKI Services home page. (See “#unique_66/unique_66_Connect_42_mainpage” on page 376.)

____

Relationship between certificate requests and matching certificates

PKI Services maintains two databases:

- The request database (RDB), also called the object store
- The issued certificate list (ICL)

RDB records are temporary in nature. They exist only to track active requests. PKI Services automatically removes these records when they are complete or go inactive. ICL records are permanent. Requests for certificates (both new and renewal) are stored in the RDB. Once approved, a matching certificate is created from the request and stored in the ICL. (Note, the creation of the certificate might not be instantaneous.) At this point, the two database records, though related, exist independently of each other.

- After a request is approved, there is no way for you to *un*-approve a request. If you mistakenly approve a request that you meant to reject, you should immediately delete the RDB entry. This prevents the user from retrieving the certificate. You should then search the issued certificates to see if the certificate has been issued. If it has, you should revoke it in case the user has already picked it up.
- Revoking a certificate (an ICL action) has no effect on its matching RDB entry. If you revoke a certificate, you should also delete its matching RDB entry if it exists. This prevents the user from retrieving the certificate, if the user has not already done so.
- You can delete RDB entries any time after they have been completed to save space in the database if you want.
- Under normal circumstances, ICL entries should not be deleted. If you delete an ICL entry, you are no longer able to revoke or renew the certificate.
- You can delete entries in any state in either database to clean up error conditions.

Chapter 20. Using PKI Services utilities

This topic describes the following utility programs, which are shipped with PKI Services. These programs are installed in the `/`*`install-dir`*`/pkiserv/bin` directory.

createcrls
: A UNIX program that initiates the certificate revocation list (CRL) creation task immediately.

db2conv
: A UNIX program that converts the version format of object store and ICL Db2 tables.

iclview
: A UNIX program that displays the entries in the issued certificate list (ICL).

pkiprereg
: A UNIX program that creates Simple Certificate Enrollment Protocol (SCEP), Enrollment over Secure Transport (EST), or preregistration records

postcerts
: A UNIX program that creates posting objects for existing certificates. The PKI Services daemon later posts the certificates to an LDAP server.

TemplateTool
: A Java program that validates an XML certificate template file and converts it to a text CGI template file, and converts a text CGI template file to an XML template file.

vosview
: A UNIX program that displays the entries contained in the object store (request database).

vsam2db2
: A UNIX program that converts data from the issued certificate list (ICL) and object store VSAM data sets into Db2 tables.

vsamconv
: A UNIX program that converts the version format of object store and ICL VSAM data sets.

Using the createcrls utility

Purpose

The `createcrls` program initiates the task that creates certificate revocation lists (CRLs). Later, depending on how PKI Services is configured, the PKI Services daemon either posts the CRLs to an LDAP server (for LDAP) or saves them in the HFS (for the URI format). (The `PostInterval` parameter in the **LDAP** section of the configuration file determines when the posting to LDAP occurs.) You can use this program to create a CRL immediately, instead of waiting for PKI Services to do it automatically based on the `TimeBetweenCRLs` parameter in the configuration file.

Path setup

Update your PATH, LIBPATH, and NLSPATH environment variables with the appropriate `pkiserv` directory before you run `createcrls`. (Note that you are updating the environment variables for the user running the utility, not updating values in the PKI Services environment variables file, `pkiserv.envars`.) Once you have updated these variables, you can run `createcrls` from the UNIX command line.

Variable name	You must add ...
PATH	`/`*`install-dir`*`/pkiserv/bin`
LIBPATH	`/`*`install-dir`*`/pkiserv/lib`
NLSPATH	`/`*`install-dir`*`/pkiserv/lib/nls/msg/%L/%N`

The default directory for *install-dir* is `/usr/lpp`.

Format

```
createcrls [-D CA-Domain-name]
```

Parameters

-D *CA-Domain-name*
: Specifies the 1-8 character name of the CA domain for which CRLs are to be created. The name can be entered using uppercase or lowercase letters. This option is required only if PKI Services is running with multiple CA domains.

Examples

To create a CRL for the domain `mydomain` and post the CRL to LDAP, enter the command:

```
createcrls –D mydomain
```

To create a CRL and post it to LDAP if you are not running PKI Services with multiple CA domains, enter the command:

```
createcrls
```

Using the db2conv utility

Purpose

The db2conv utility is used to convert Db2 object store and ICL from `version 0` format to `version 1` format, or from `version 1` format to `version 0` format. `Version 1` format is introduced in z/OS Version 2 Release 3. Run `db2conv` when PKI Services is not running.

Path setup

Update your PATH, LIBPATH, and NLSPATH environment variables with the appropriate `pkiserv` directory before you run `db2conv`. (Note that you are updating the environment variables for the user running the utility, not updating values in the PKI Services environment variables file, `pkiserv.envars`.) After you update these variables, you can run `db2conv` from the UNIX command line.

Variable name	You must add ...
PATH	`/`*`install-dir`*`/pkiserv/bin`
LIBPATH	`/`*`install-dir`*`/pkiserv/lib`
NLSPATH	`/`*`install-dir`*`/pkiserv/lib/nls/msg/%L/%N`

The default directory for *install-dir* is `/usr/lpp`.

Format

```
db2conv [-h] -s pkg=source-db2-package-name,ver=version
         -t pkg=target-db2-package-name
         -a
         -b db2-subsystem-name
```

Parameters

-h
Displays the help message.

-s pkg=*source-db2-package-name*,ver=*version*
Specifies the source Db2 information with the package name and version in the following format:

```
pkg=package-name,ver=version
```

version

0
Specifies tables without the SCEP transaction ID column and version column.

1
Specifies tables with the SCEP transaction ID column and version column.

-t pkg=*target-db2-package-name*
Specifies the target Db2 information in the following format:

```
pkg=package-name
```

-a
If specified, all records from the source object store (OST) and issued certificate list (ICL) are converted to form records in the target object store and issued certificate list. If not specified:

- For the object store, all objects except the completed certificate request objects are converted.
- For the ICL, all certificate objects except the expired certificates for which the user provided the public key, are converted.

Note: If numerous completed certificate requests and the expired certificates that are created with user-supplied keys exist, it can take longer to run the utility with the `-a` option.

-b *db2-subsystem-name*
Specifies the Db2 subsystem name.

Examples

This sample script performs operations on the tables in the DSN9 Db2 subsystem by using two packages. The source package is `MasterCA` and the target package is `MasterV1`. It converts the records from the `version 0` object store and ICL tables, to the `version 1` tables. All of the records in the Db2 tables are converted, including those for completed requests and expired certificates.

```
db2conv -b DSN9 -s pkg=MasterCa,ver=0 -t pkg=MasterV1 -a
```

Output

As the `db2conv` utility runs, it reports its progress. For every 1000 records that are converted from VSAM data sets to Db2 tables, the utility reports the total number of records converted.

If an error occurs, `db2conv` displays a count of processed records and a count of converted records.

If the source package and the version do not match, an error is issued.

Using the iclview utility

Purpose

The `iclview` program displays the entries contained in an issued certificate list (ICL). Depending on how you have configured PKI Services, the ICL can be in a VSAM data set or in a Db2 table. Each ICL record consists of a fixed header followed by a variable-length section containing the BER-encoded certificate.

For each entry `iclview` displays the header information and optionally calls a user-provided program to process the BER-encoded certificate.

Path setup

Update your PATH, LIBPATH, and NLSPATH environment variables with the appropriate `pkiserv` directory before you run `iclview`. (Note that you are updating the environment variables for the user running the utility, not updating values in the PKI Services environment variables file, `pkiserv.envars`.) Once you have updated these variables, you can run `iclview` from the UNIX command line.

Variable name	You must add ...
PATH	/*install-dir*/pkiserv/bin
LIBPATH	/*install-dir*/pkiserv/lib
NLSPATH	/*install-dir*/pkiserv/lib/nls/msg/%L/%N

The default directory for *install-dir* is `/usr/lpp`.

Format

```
iclview {-d vsam-dataset-name [-r] | -b db2-subsystem-name -k db2-package-name
        | -v ICL-version
        | -c [-p path]}
        [-D CA-domain-name] [-s decode-command-string]
```

Parameters

You can display usage information about the `iclview` command format and parameters when you issue the `iclview` utility command with no parameters.

-d *vsam-dataset-name*
Specifies the MVS data set name of the VSAM issued certificate list (ICL).

If you specify `-d`, do not specify `-b`, `-k` or `-c`. If you do, the utility issues an error message and the command fails.

Note: If the data set name has no quotes, the program uses the invoker of the command as the first qualifier. If you specify the fully-qualified data set name, use quotes, and make sure to include the *escape* character, which is a backslash (\), before the quotation marks enclosing the data set name. For example, see `\'pkisrvd.vsam.ost\'` in "Examples" on page 446.

-r
Indicates to open in VSAM record-level sharing (RLS) mode the VSAM data set specified with the `-d` option.

`-r` is ignored and the utility issues a warning message if `-b` and `-k` are specified, or if `-c` is specified.

-b *db2-subsystem-name*
Specifies the name of the Db2 subsystem or group attachment where the ICL is located.

If you specify `-b` and you do not specify `-k`, or you specify `-d` or `-c`, the utility issues an error message and the command fails.

-k *db2-package-name*
Specifies the Db2 package name of the ICL.

If you specify `-k` and do not specify `-b`, or you specify `-d` or `-c`, the utility issues an error message and the command fails.

-v *ICL-version*
Specifies the ICL version that corresponds to the *db2-package-name* specified by the `-k` option. It will be ignored if `-k` is not specified. The acceptable value is 0 or 1.

If you specify `-k` and do not specify `-v`, the version defaults to 0.

-c
Indicates to retrieve the location of the ICL from the `pkiserv.conf` configuration file. Either the VSAM data set name is retrieved, or the Db2 subsystem name and package name are retrieved, depending which you are using. For VSAM, the `SharedPlex` value determines whether the VSAM data set is opened in record-level sharing (RLS) mode. (For Db2, the `SharedPlex` value has no effect on this utility.)

If you specify `-c`, and you also specify `-d`, `-b`, or `-k`, the utility issues an error message and the command fails.

Note: When you also specify the `-D` option, you must use the `-p` option to specify the CA domain configuration directory if it is not `/etc/pkiserv`.

-p *path*
Specifies the directory where the `pkiserv.conf` configuration file resides. If not specified, the directory defaults to `/etc/pkiserv`. This option is only valid when specified with the `-c` option. If specified with the `-b`, `-k`, or `-d` options, the utility issues a warning message, and the `-p` option is ignored.

-D *CA-domain-name*
Specifies the CA domain name where this utility command is directed.

Notes:

1. The `-D` option is required only if PKI Services is running in multiple-CA mode.
2. The *CA-domain-name* value can be entered using uppercase or lowercase letters.
3. When you also specify the `-c` option, you must use the `-p` option to specify the CA domain configuration directory if not `/etc/pkiserv`.

-s *decode-command-string*
Specifies an optional command to call for decoding the ASN.1-encoded data. The command must be able to read and decode binary (BER) data from STDIN.

Examples

To view the records in the ICL in the VSAM data set `'PKISRVD.VSAM.ICL'`, passing the certificate to a utility called `dumpasn1`, use the following command:

```
iclview -d \'pkisrvd.vsam.icl\' -s 'dumpasn1 -'
```

To view the records in the ICL for the CA domain MasterCA using the information from the `pkiserv.conf` file located in the directory `/etc/pkiserv/MasterCA`, use the following command:

```
iclview -c -p /etc/pkiserv/MasterCA -D MasterCA
```

To view the records in the ICL in the Db2 subsystem DSN9 with a package name of MasterCA, passing the certificate to a utility called `dumpasn1` and directing the output to the file `icl.out`, use the following command:

```
iclview -b DSN9 -k MASTERCA -s 'dumpasn1 -' >icl.out
```

Note: A `dumpasn1` utility is not shipped with PKI Services.

Output

The fixed header data that is displayed for each record is similar to the following sample:

```
Cert 8: John Q. Public@someWebProvider.com
     ISSUED (Issued certificate)
     Issued at 2017-12-19 17:27:41
     Last changed 2017-12-19 17:42:30
     Subject: CN=John Smith,OU=Class 1 Internet Certificate CA,O=The Firm
     Issuer: OU=PKI Services CA,O=IBM,C=US
     Requester: John Smith
     ApplData: 5YSCEPP
     Serial Number: 15
     Email flag: Off
     AutoRenew flag: Enabled
     Additional flags Set: NotRenewable
     KeyID: 12FD68977EE1F987DC9CA1440B62CCCD1CD0A9BB
     Validity: 2017/07/19 00:00:00 – 2022/07/18 23:59:59
     Revocation information location: Distribution Point 151
     SCEP TID: EE7216D9E9B8B0EB74C3897D57E79662E750E06496A7B4A92A2789FD3EE1C41F
```

The first line of the output specifies the certificate's sequential position within the ICL, relative to the other certificates, and the requestor's name.

The second line specifies the certificate state, which is one of the following states, and a comment (if any):

- ISSUED
- REVOKED, not posted
- REVOKED, awaiting CRL post
- REVOKED, on posted CRL

Issued at
Indicates when the certificate was issued.

Last changed
Indicates when the administrator last changed the certificate.

Subject
Indicates the name of the person who owns the certificate.

Issuer
Indicates the name of the certificate authority that issued the certificate.

Requester
Indicates the requester's name.

Appldata
Indicates the 8-character string identifying to the application the short name or nickname of the certificate template. (PKI Services provides sample certificate templates but it is RACF, or an equivalent security product, rather than PKI Services, that handles the SAF templates.) Table 36 on page 152 shows the nicknames for each certificate template. (These nicknames are supplied in the `pkiserv.tmpl` certificate templates file as defaults but your installation might have changed them or added others during customization. See "TEMPLATE sections" on page 151 for more information.)

Serial Number
Indicates the serial number of the certificate as a hexadecimal number.

Email flag
Indicates whether to send an expiration warning message. The possible values are On or `Off`.

AutoRenew flag
Indicates whether a certificate is AutoRenew enabled, disabled, or not set up for automatic renewal.

Revoked at
Indicates the date and time the certificate was revoked or suspended. This line is only displayed if the certificate is revoked.

Revocation Reason
The following are potential reasons for the revocation of the certificate:

- No reason.
- User key was compromised.

- Original use no longer valid.
- CA key was compromised.
- User changed affiliation.
- Certificate was superseded.
- Temporarily suspended.

Additional flags Set
The NotRenewable flag indicates that the certificate cannot be renewed. This state occurs when the keys for the certificate were generated by PKI Services, the certificate has a Mail RDN, and the user's email address has changed since the certificate was created.

KeyID
The SHA1 hash of the public key in EBCDIC format, 40 bytes in length. The value of the **KeyID** is displayed if the key was generated by PKI Services, otherwise, a value is not displayed.

Validity
The validity period of the certificate.

Revocation information location
The location of the revocation information of the end entity certificate. This location can be different than the location specified in the CRLDistributionPoint extension in the certificate if the CRLDistName value was changed after the certificate was created. This line is displayed only if there is revocation information and it can be retrieved successfully. The information can be a distribution point number, or "Master CRL".

Note: CA certificates that are issued by the PKI Services CA are always placed in the master authority revocation list (ARL) and the ARL distribution point if configured. Therefore, the revocation information location is not displayed for issued CA certificates.

SCEP TID
The SCEP transaction ID data. This line is only displayed if it is a certificate that is originated from a SCEP request.

Using the pkiprereg utility

Purpose

The `pkiprereg` program creates Simple Certificate Enrollment Protocol (SCEP) and Enrollment over Secure Transport (EST) preregistration records in batch.

Path setup

Update your PATH, LIBPATH, and NLSPATH environment variables with the appropriate `pkiserv` directory before you run `pkiprereg`. (Note that you are updating the environment variables for the user running the utility, not updating values in the PKI Services environment variables file, `pkiserv.envars`.) Once you have updated these variables, you can run `pkiprereg` from the UNIX command line.

Variable name	You must add ...
PATH	`/`*`install-dir`*`/pkiserv/bin`
LIBPATH	`/`*`install-dir`*`/pkiserv/lib`
NLSPATH	`/`*`install-dir`*`/pkiserv/lib/nls/msg/%L/%N`

The default directory for *install-dir* is `/usr/lpp`.

Format

```
pkiprereg {-m mode [-t tmpl-file] -s SCEP-file [-o out-file] [-d domain]
          | [-h | -?]}
```

Parameters

You can display usage information about the `pkiprereg` command format and parameters when you execute the `pkiprereg` utility command with the `-h` or `-?` option or when you enter an incorrect parameter.

`-m` *mode*
Indicates one of the following modes of operation for this `pkiprereg` utility execution.

Mode name	Function
verify	Checks the data file for format errors but does *not* load the records into PKI Services.
generate	Generates a random 8-character passphrase whenever it finds a passphrase *placeholder* (the `*` character) in the file. It does *not* load the records into PKI Services.
load	Calls PKI Services to load the preregistration records.
remove	Calls PKI Services to remove the preregistration records.

You can specify the mode value as the mode name or any number of initial characters of the mode name. **Example:** You can specify **verify** mode using any of the following options:

- `-m verify`
- `-m ve`
- `-m v`

`-t` *tmpl-file*
When used in **verify** or **load** mode, specifies the path name of the PKI Services certificate templates file (`pkiserv.tmpl` if you are implementing the web application using REXX CGI scripts, or `pkixgen.tmpl` if you are implementing the web application using JavaServer pages), which is used as input. When used in other modes, the `-t` option is ignored.

`-s` *SCEP-file*
Specifies the path name of the file containing the preregistration data that is input for all modes.

`-o` *out-file*
When used in **generate** mode, specifies the path name of the output file for the preregistration data. When used in other modes, the `-o` option is ignored.

`-d` *domain*
When used in **load** or **remove** mode, specifies the PKI Services CA domain name where this utility command is directed.

Notes:

1. When used in other modes, the `-d` option is ignored.
2. The `-d` option is required only if PKI Services is running in multiple-CA mode.
3. The *domain* value can be entered using uppercase or lowercase letters.

`-h` or `-?`
Displays the syntax of the `pkiprereg` command.

Input

The input for the `pkiprereg` utility is a data file (*SCEP-file*) containing preregistration records. The rules regarding format of the preregistration records are as follows:

Rules:

- Each preregistration record consists of multiple consecutive *name=value* pairs, terminated by a blank line or an end-of-file indicator.
- Each line consists of single field name and its value, separated by a = character, forming one *name=value* pair per line.
- Each *name=value* pair (except the Template *name=value* pair) represents data that the client must supply at enrollment time for authentication purposes.
- In `pkiprereg` **verify** and **load** modes, the Template and ClientName *name=value* pairs are required. All other *name=value* pairs are optional.
- In `pkiprereg` **remove** mode, only the ClientName *name=value* pair is required.
- Any line beginning with a # character is considered a comment.
- All characters before the = character, except any leading and trailing white space, are considered the *name*.
- All characters after the = character, except any leading and trailing white space, are considered the *value*.
- The field name supplied as a *name* is case-sensitive and must match one of the field names listed in Table 75 on page 439.
- Line length is limited to 300 characters. After 300 characters, any additional characters are truncated.

The following field names are supported in the preregistration record.

Table 75. List of valid field names for use in the preregistration record as input to the pkiprereg utility

Field name	**Maximum length**	**Description**
Template	8 characters	The nickname of the preregistration template to be used.
ClientName	64 characters	The name of the person or device being preregistered. The first 32 characters are case-insensitive and must be unique for each user preregistered in PKI Services.
PassPhrase	32 characters	The password to be communicated to the requestor. The `*` value can be used as a *placeholder* when running in **generate** mode.
SerialNumber	64 characters	Serial number of the subject device.
UnstructAddr	64 characters	Unstructured address of the subject device.
EmailAddr	64 characters	Email address for the `EMAIL=` attribute for subject distinguished name.
Mail	64 characters	Email address for the `MAIL=` attribute for subject distinguished name.
DNQualifier	64 characters	Specifies information to add to the subject's distinguished name to make it unambiguous.
Uid	64 characters	The system login name associated with the subject.
Title	64 characters	Title for subject distinguished name.
DomainName	64 characters	One component of a domain name. For example, domain name www.ibm.com is represented by 3 components: www, ibm, and com.
OrgUnit	64 characters	Organizational unit for subject distinguished name.
Org	64 characters	Organization for subject distinguished name.
Street	64 characters	Street for subject distinguished name.
Locality	64 characters	Locality for subject distinguished name.

Table 75. List of valid field names for use in the preregistration record as input to the pkiprereg utility (continued)

Field name	Maximum length	Description
StateProv	64 characters	State or province for subject distinguished name.
PostalCode	64 characters	Postal code for subject distinguished name.
Country	2 characters	Country abbreviation for subject distinguished name.
AltIPAddr	45 characters	The IP address for the subject alternate name extension. PKI Services supports IP version 4 and IP version 6 addresses. • For IP version 4, the IP address is in dotted decimal format; for example, `9.67.97.103`. • For IP version 6, the IP address is divided into eight 16-bit hexadecimal blocks separated by colons. Leading zeros in each 16-bit field are optional, and successive fields of zeros can be represented by double colons, but only once; for example `1:2::3:4` is equivalent to `0001:0002:0000:0000:0000:0000:0003:0004`. • In a mixed IP version 4 and IP version 6 environment, the IP address can be expressed in the format *x:x:x:x:x:x:d.d.d.d,* where the *x* values are the hexadecimal values of the six high-order 16-bit pieces of the address, and the *d* values are the decimal values of the four low-order 8-bit pieces of the address in standard IP version 4 representation; for example, `0:0:0:0:0:ABCD:1.2.3.4`, or the equivalent value `::ABCD:1.2.3.4`
AltURI	255 characters	URI for subject alternate name extension.
AltEmail	100 characters	Email address for subject alternate name extension.
AltDomain	100 characters	Domain name for subject alternate name extension.
AltOther	255 characters	Other Name for subject alternate name extension.

The following field names are supported as input to the `pkiprereg` utility, but they are not intended for use with SCEP certificates.

Table 76. List of valid field names for use in the preregistration record as input to the pkiprereg utility but not intended for use with SCEP certificates

Field name	Maximum length	Description
BusinessCat	64 characters	Business category for subject distinguished name.
JurCountry	2 bytes	Jurisdiction country for subject distinguished name.
JurStateProv	64 characters	Jurisdiction state/province for subject distinguished name.
JurLocality	64 characters	Jurisdiction locality for subject distinguished name.

For information about the format required for these values, see the `R_PKIServ` (IRRSPX00) section of *z/OS Security Server RACF Callable Services*.

Examples

The following example shows two preregistration records in the `/etc/scep.txt` file.

```
Template=5YSCEPP
ClientName=www.ibm.com
PassPhrase=gumby
Org=IBM
Template=5YSCEPP
ClientName=scep.company.com
PassPhrase=*
```

The following command examples produce the actions listed:

Example 1:

```
pkiprereg -m v -t /etc/pkiserv/pkiserv.tmpl -s /etc/scep.txt
```

Action: Verifies the `/etc/scep.txt` file, indicating that one PassPhrase has an incorrect value, the placeholder (*).

Example 2:

```
pkiprereg -m g -s /etc/scep.txt -o /etc/scepfinal.txt
```

Action: Generates a passphrase for the placeholder and saves the records in the `/etc/scepfinal.txt` output file.

Example 3:

```
pkiprereg -m l -t /etc/pkiserv/pkiserv.tmpl -s /etc/scepfinal.txt
```

Action: Loads the preregistration records into PKI Services. If the template identified by the nickname 5YSCEPP contains any subject or alternate name information in the **<CONSTANT>** section, that information overrides any matching information specified in the `/etc/scepfinal.txt` file.

Using the postcerts utility

Purpose

The `postcerts` program creates LDAP posting objects for certificates, which the PKI Services daemon later posts to an LDAP directory. (The `PostInterval` parameter in the **LDAP** section of the configuration file determines when the posting occurs.) You can use this utility if you have created certificates that PKI Services did not automatically post to an LDAP directory; for example if you created certificates before you configured PKI Services to automatically post them.

Path setup

Update your PATH, LIBPATH, and NLSPATH environment variables with the appropriate `pkiserv` directory before you run `postcerts`. (Note that you are updating the environment variables for the user running the utility, not updating values in the PKI Services environment variables file, `pkiserv.envars`.) After you have updated these variables, you can run `postcerts` from the UNIX command line.

Variable name	You must add ...
PATH	`/`*`install-dir`*`/pkiserv/bin`
LIBPATH	`/`*`install-dir`*`/pkiserv/lib`
NLSPATH	`/`*`install-dir`*`/pkiserv/lib/nls/msg/%L/%N`

The default directory for *install-dir* is `/usr/lpp`.

Format

```
postcerts –s serial-numbers [-D CA-domain-name] [-c comment]
```

Parameters

–s *serial-numbers*
The serial numbers of the certificates to be posted. This parameter contains one or more serial number specifiers, which are separated by commas, where a serial number specifier can be:

- A single serial number
- Two serial numbers that are separated by a dash, indicating a range of serial numbers

-D *CA-domain-name*
The 1-8 character name of the CA domain for which certificates are to be posted. The name can be entered using uppercase or lowercase letters. This option is required only if PKI Services is running with multiple CA domains.

-c *comment*
A comment to be stored in the issued certificate list (ICL).

Examples

To post a single certificate with serial number X'17':

```
postcerts –s 17
```

To post multiple certificates with a comment:

```
postcerts –s 17,18,8A –c ‘Posting Certificates for Coop’
```

To post certificates whose serial numbers are in the range X'20' to X'3B':

```
postcerts –s 20-3B
```

To post certificates whose serial numbers are in the ranges X'20' to X'3B' and to also post a single certificate with serial number X'17':

```
postcerts –s 20-3B,17
```

Using the TemplateTool utility

Purpose

The TemplateTool utility works with certificate template files. It performs functions that you need if you implement the web application using JavaServer pages (JSPs). There are three certificate template files that are used by PKI Services:

- A text CGI template file, `pkiserv.tmpl`, used to implement the web application using REXX CGI execs.
- An XML template file, `pkitmpl.xml`, used to implement the web application using JavaServer pages (JSPs).
- A text CGI template file, `pkixgen.tmpl`, generated from the XML template file, and used by the PKI Services daemon if you implement the web application using JSPs. Any time that you update `pkitmpl.xml`, you must use TemplateTool to create an equivalent copy of `pkixgen.tmpl`. If you do not do this, the daemon writes the following message to the daemon log file when it determines that `pkixgen.tmpl` is not current:

  ```
  IKYC068I The templates file used may not be current
  ```

The TemplateTool utility performs the following functions:

- validateXML

  The validateXML function validates an XML template file against the XML schema file. It can optionally convert the XML template file to a text CGI template file. Use this function after you make changes to your XML schema file, `pkiserv.tmpl`, to validate the changes and create a CGI template file, `pkixgen.tmpl`, for use by the PKI Services daemon.
- convertCGI

  The convertCGI function converts a text CGI template file to an XML template file. This function can assist PKI Services administrators in converting the web application from the REXX CGI execs to the JSPs. The conversion process might not convert all tags, so you should always verify the contents of the resulting XML template file.
- help

  The help function prints the format of the utility.

Requirement: You must have Java V5.0 or later installed to run the TemplateTool utility.

Path setup

Update your PATH and CLASSPATH environment variables with the appropriate directories before you run `TemplateTool`. (Note that you are updating the environment variables for the user running the utility, not updating values in the PKI Services environment variables file, `pkiserv.envars`.) When you have updated these variables, you can run `TemplateTool` from the shell script or from Java.

Variable name	You must add ...
PATH	`/`*`install-dir`*`/pkiserv/bin`
	The directory containing the `java` command.
CLASSPATH	`/`*`install-dir`*`/pkiserv/lib/pki_xml.jar`

The default directory for *install-dir* is `/usr/lpp`.

Format

```
TemplateTool -help |

             -validateXML [-XMLTemplate XML_template_file]
                          [-CGITemplate CGI_template_file]  |

             -convertCGI [-CGITemplate CGI_template_file]
                         [-XMLTemplate XML_template_file]
```

To run the utility from Java:

```
java com.ibm.pki.template.TemplateTool function parameters
```

To run the utility from the shell script:

```
TemplateTool function parameters
```

Parameters

-XMLTemplate *XML_template_file*

An XML template file. Specify the name of the XML template file if it is in the current directory, or the full path name.

For the validateXML function, this parameter is required. The specified XML template file is validated against the XML schema file `PKIServ.xsd`, which must be in the same directory.

For the convertCGI function, this parameter is required. The CGI template file specified as input is converted to an XML template file using the XML schema file `PKIServ.xsd`, and output to the file specified by -XMLTemplate.

-CGITemplate *CGI_template_file*

A CGI template file. Specify the name of the CGI template file if it is in the current directory, or the full path name.

For the validateXML function, this parameter is optional. If specified, the XML template file specified as input is converted into CGI format and output to the file specified by -CGITemplate. If the CGI template file is to be used by the PKI Services daemon, it must be named `pkixgen.tmpl`.

For the convertCGI function, this parameter is required. The CGI template file specified is converted to an XML template file using the XML schema file `PKIServ.xsd`, which must be in the same directory, and output to the file specified by -XMLTemplate.

Examples

Example 1:

```
TemplateTool -validateXML -XMLTemplate pkitmpl.xml -CGITemplate pkixgen.tmpl
```

Action: Invokes TemplateTool from the shell script. Validates that the XML template file `pkitmpl.xml` is valid, and generates a text CGI template file from it named `pkixgen.tmpl`, for use by the PKI Services daemon.

Example 2:

```
java com.ibm.pki.template.TemplateTool -convertCGI -CGITemplate pkiserv.tmpl
-XMLtemplate pkitmpl.xml
```

Action: Invokes TemplateTool from Java. Converts the text CGI template file named `pkiserv.tmpl` to an XML template file named `pkitmpl.xml`.

Using the vosview utility

Purpose

The `vosview` program displays the data in the PKI Services object store (the request database). Depending on how you have configured PKI Services, the object store can be in a VSAM data set or in a Db2 table. Each request record consists of a fixed header, followed by a variable-length section. For each entry `vosview` displays the header information and optionally calls a user-provided program to process the BER-encoded request.

Path setup

Update your PATH, LIBPATH, and NLSPATH environment variables with the appropriate `pkiserv` directory before you run `vosview`. (Note that you are updating the environment variables for the user running the utility, not updating values in the PKI Services environment variables file, `pkiserv.envars`.) Once you have updated these variables, you can run `vosview` from the UNIX command line.

Variable name	You must add ...
PATH	`/`*`install-dir`*`/pkiserv/bin`
LIBPATH	`/`*`install-dir`*`/pkiserv/lib`
NLSPATH	`/`*`install-dir`*`/pkiserv/lib/nls/msg/%L/%N`

The default directory for *install-dir* is `/usr/lpp`.

Format

```
vosview {-d vsam-dataset-name [-r] | -b db2-subsystem-name -k db2-package-name
        | -v ObjectStore-version
        | -c [-p path]}
        [-D CA-domain-name] [-s decode-command-string]
```

Parameters

You can display usage information about the `vosview` command format and parameters when you issue the `vosview` utility command with no parameters.

-d *vsam-dataset-name*
Specifies the MVS data set name of the VSAM object store.

If you specify `-d`, do not specify `-b`, `-k` or `-c`. If you do, the utility issues an error message and the command fails.

Note: If the data set name has no quotes, the program uses the invoker of the command as the first qualifier. If you specify the fully qualified data set name, use quotes, and make sure to include the *escape* character, which is a backslash (\), before the quotation marks enclosing the data set name. For example, see `\'pkisrvd.vsam.ost\'` in "Examples" on page 446.

-r
Indicates to open in record-level sharing (RLS) mode the VSAM data set specified with the `-d` option.

`-r` is ignored and the utility issues a warning message if `-b` and `-k` are specified, or if `-c` is specified.

-b *db2-subsystem-name*
Specifies the name of the Db2 subsystem or group attachment where the object store is located.

If you specify `-b` and you do not specify `-k`, or you specify `-d` or `-c`, the utility issues an error message and the command fails.

-k *db2-package-name*
Specifies the Db2 package name of the object store.

If you specify `-k` and do not specify `-b`, or you specify `-d` or `-c`, the utility issues an error message and the command fails.

-v *ObjectStore-version*
Specifies the object store version that corresponds to the *db2-package-name* specified by the `-k` option. It is ignored if `-k` is not specified. The acceptable value is 0 or 1.

If you specify `-k` and do not specify `-v`, the version defaults to 0.

-c
Indicates to retrieve the location of the object store from the `pkiserv.conf` configuration file. Either the VSAM data set name is retrieved, or the Db2 subsystem name and package name are retrieved, depending which you are using. For VSAM, the `SharedPlex` value determines whether the VSAM data set is opened in record-level sharing (RLS) mode. (For Db2, the `SharedPlex` value has no effect on this utility.)

If you specify `-c`, and you also specify `-d`, `-b`, or `-k`, the utility issues an error message and the command fails.

Note: When you also specify the `-D` option, you must use the `-p` option to specify the CA domain configuration directory if it is not `/etc/pkiserv`.

-p *path*
Specifies the directory where the `pkiserv.conf` configuration file resides. If not specified, the directory defaults to `/etc/pkiserv`. This option is only valid when specified with the `-c` option. If specified with the `-b`, `-k`, or `-d` options, the utility issues a warning message, and the `-p` option is ignored.

-D *CA-domain-name*
Specifies the CA domain name where this utility command is directed.

Notes:

1. The `-D` option is required only if PKI Services is running in multiple-CA mode.
2. The *CA-domain-name* value can be entered using uppercase or lowercase letters.
3. When you also specify the `-c` option, you must use the `-p` option to specify the CA domain configuration directory if not `/etc/pkiserv`.

-s *decode-command-string*
Specifies an optional command to call for decoding the ASN.1-encoded data. (The command must be able to read and decode binary (BER) data from STDIN.)

Examples

To view the records in the VSAM object store data set `'PKISRVD.VSAM.OST'`, passing the request data to a utility called `dumpasn1`, use the following command:

```
vosview -d \'pkisrvd.vsam.ost\' -s 'dumpasn1 -'
```

To view the records in the object store for the CA domain MasterCA using the information from the `pkiserv.conf` file located in the directory `/etc/pkiserv/MasterCA`, use the following command:

```
vosview -c -p /etc/pkiserv/MasterCA -D MasterCA
```

To view the records in the object store in the Db2 subsystem DSN9 with a package name of MasterCA, passing the request data to a utility called `dumpasn1`, and redirecting the output to the file `vos.out`, use the following command:

```
vosview -b DSN9 -k MASTERCA -s 'dumpasn1 -' >vos.out
```

Note: A `dumpasn1` utility is not shipped with PKI Services.

Output

Records with an object key value of 100 or higher display common information stored for each record. Records with an object key value less than 100 are special records maintained by the PKI Services daemon and the information displayed is specific to the record.

Sample record 1

```
Object key = 1
Last used key = 110, CRL serial number = 2, ARL serial number =  2
   High DP = 2, Low DP = 1
Creation time is: 2017/03/09 14:46:49
Last modified time is: 2017/03/09 14:46:49
```

Last used key
The primary index for the last record in the data set.

CRL serial number
The number to be used for the next CRL.

ARL serial number
The number to be used for the next ARL.

High DP
The number of the highest distribution point CRL issued by PKI Services.

Low DP
The number of the lowest currently active distribution point CRL.

Sample record 2

```
Object key = 2
The next CRL event is scheduled for 2017/03/09 at 14:50:53
Creation time is: 2017/03/09 14:37:17
Last modified time is: 2017/03/09 14:45:53
```

Sample record 3

```
Object key = 3
The CRL Distribution Point name in effect is: CN=CRL%u,OU=Master CA,O=IBM,C=US
(Note: A CRL DP number is substituted for the %u in the CRL DP name above)
Creation time is: 2017/03/09 14:37:17
Last modified time is: 2017/03/09 14:37:17
```

Sample certificate request record

```
Object key = 108
name = "fips0 selfenroll 1024sha256"
tid = 1lg0ffEhCyC+2SHV++++++++
appldata = 5YSCEPP
comment = AuthenticatedClient= AutoApprove
data len = 1659
flags = 2022010 - Type = Cert    State = CA CertSigned    Complete
Creation time is: 2017/03/09 14:46:03
Last modified time is: 2017/03/09 14:46:03
SCEP tid = EE7216D9E9B8B0EB74C3897D57E79662E750E06496A7B4A92A2789FD3EE1C41FA8AEB
```

Object key
The index into the VSAM data set name.

name
The requestor's name.

tid
The transaction ID data.

appldata
Indicates the 8-character string identifying to the application the short name or nickname of the certificate template. (PKI Services provides sample certificate templates but it is RACF, or an equivalent security product, rather than PKI Services, that handles the SAF templates.) Table 36 on page 152 shows the nicknames for each certificate template. (These nicknames are supplied in the `pkiserv.tmpl` certificate templates file as defaults but your installation might have changed them or added others during customization. See "TEMPLATE sections" on page 151 for more information.)

comment
A comment the administrator supplied the last time that the request was updated.

data len
The length of the variable data portion (that is, the BER-encoded request).

flags
Represent the current state of the request:

Type

Cert
Certificate request (new or renewal).

CRL
Certificate revocation list (CRL).

Rev
Revocation request.

Post
Certificate waiting to be posted to LDAP.

State
The prefix (RA or CA) and one of the following values:

CertPreregistered
Certificate preregistration record.

CertReqActive
Certificate request in some state of being completed.

CertSigned
Certificate request where the certificate has been created.

CertReqRejected
Certificate request that has been rejected.

RevReqActive
Revocation request in some state of being completed.

CRLWaitingForRA
CRL to be posted to LDAP.

CertPostPending
Certificate to be posted to LDAP.

CaInfoPostPending
PKI Services' CA certificate to be posted to LDAP.

State Flag
Optional. If present, is one of the following values:

Complete
Request is complete. For approved requests, the end user has retrieved the certificate.

Error
The certificate could not be posted to LDAP.

NeedsConfirm
Approved or rejected. End user has yet to be notified of the outcome.

AutoRenewEnabled
The certificate returned automatic renewal and this capability is enabled.

AutoRenewCapable
The certificate returned active certificates capable for auto renewal but disabled.

Synchronous
Request is an in-progress synchronous certificate request.

SCEP tid
The SCEP transaction ID data. This line is only displayed if it is a certificate originated from a SCEP request.

Using the vsam2db2 utility

Purpose

The `vsam2db2` program copies data from the issued certificate list (ICL) and object store VSAM data sets into Db2 tables. Use this utility if you have been using VSAM data sets for the ICL and object store and want to use Db2 tables instead. Run `vsam2db2` when PKI Services is not running. You must create the Db2 tables before you run `vsam2db2`. Ensure that the VSAM version corresponds to the Db2 package. For more information, see "Converting the object store and ICL from VSAM to Db2" on page 125.

Path setup

Update your PATH, LIBPATH, and NLSPATH environment variables with the appropriate `pkiserv` directory before you run `vsam2db2`. (Note that you are updating the environment variables for the user running the utility, not updating values in the PKI Services environment variables file, `pkiserv.envars`.) When you have updated these variables, you can run `vsam2db2` from the UNIX command line.

Variable name	You must add ...
PATH	/*install-dir*/pkiserv/bin
LIBPATH	/*install-dir*/pkiserv/lib
NLSPATH	/*install-dir*/pkiserv/lib/nls/msg/%L/%N

The default directory for *install-dir* is `/usr/lpp`.

Format

```
vsam2db2 -o ostvsam-dataset-name -i iclvsam-dataset-name [-r]
         -b db2-subsystem-name -k db2-package-name
         [-D CA-domain-name] [-a]
```

Parameters

-o *ostvsam-dataset-name*
Specifies the MVS data set name of the VSAM object store.

-i *iclvsam-dataset-name*
Specifies the MVS data set name of the VSAM issued certificate list (ICL).

-r
Specifies that the VSAM data set is to be opened in record-level sharing (RLS) mode.

-b *db2-subsystem-name*
Specifies the Db2 subsystem name or the group attachment name of the object store and issued certificate list (ICL)

-k *db2-package-name*
Specifies the Db2 package name of the object store and issued certificate list (ICL)

-D *CA-domain-name*
Specifies the CA domain name

-a
If specified, all records in the object store and issued certificate list (ICL) are copied from the VSAM data sets to the Db2 tables. If not specified:

- For the object store, all objects except completed certificate request objects are copied.
- For the ICL, all certificate objects except expired certificates for which the user provided the public key are copied.

Note: It takes longer to run the utility with the `-a` option if you did not remove the completed certificate requests and the expired certificates for which the user provided the public key from the VSAM data sets.

If the version of the source object store and ICL do not match that of the target Db2 package, the error message IKYU021I is issued.

Examples

In this sample script, the object store for the domain MasterCA is in the VSAM file `pkisrvd.vsam.ost`, and the ICL is in the VSAM file `pkisrvd.vsam.icl`. They are to be copied to Db2 tables in the Db2 subsystem DSN9 and the Db2 package MasterCA. The VSAM data sets are to be opened in RLS mode. All records in the VSAM data sets are to be copied, including those for completed requests and expired certificates.

```
vsam2db2 -o \'pkisrvd.vsam.ost\' \
         -i \'pkisrvd.vsam.icl\' \
         -b DSN9 \
         -k MasterCA \
         -r \
         -D MasterCA \
         -a
```

Output

As the vsam2db2 utility runs, it reports its progress. For every 2000 records that are copied from VSAM data sets to Db2 tables, the utility reports the total number of records copied.

If an error occurs, vsam2db2 displays a count of processed records and a count of copied records. (If -a is specified, these counts should be the same.)

Using the vsamconv utility

Purpose

The vsamconv utility is used to convert the VSAM object store and ICL from version 0 format to version 1 format, or from version 1 format to version 0 format. Version 1 format is introduced in z/OS Version 2 Release 3. Run vsamconv when PKI Services is not running.

Path setup

Update your PATH, LIBPATH, and NLSPATH environment variables with the appropriate pkiserv directory before you run vsamconv. (Note that you are updating the environment variables for the user that is running the utility, not updating values in the PKI Services environment variables file, pkiserv.envars.) After you update these variables, you can run vsamconv from the UNIX command line.

Variable name	You must add ...
PATH	/*install-dir*/pkiserv/bin
LIBPATH	/*install-dir*/pkiserv/lib
NLSPATH	/*install-dir*/pkiserv/lib/nls/msg/%L/%N

The default directory for *install-dir* is /usr/lpp.

Format

```
vsamconv [-h] -s source-ostvsam-dataset-name,source-iclvsam-dataset-name [-r]
         -t target-ostvsam-dataset-name,target-iclvsam-dataset-name
         [-a]
```

Parameters

-h
: Displays the help message.

-s *source-ostvsam-dataset-name,source-iclvsam-dataset-name*
: Specifies the source VSAM object store and ICL data set.

-r
: Specifies that the VSAM data set is to be opened in record-level sharing (RLS) mode. (Optional)

-t *target-ostvsam-dataset-name,target-iclvsam-dataset-name*
: Specifies the target VSAM object store and ICL data set.

-a
: If specified, all records from the source object store and issued certificate list are converted to the target object store (OST) and issued certificate list (ICL). If not specified:

- For the object store, all objects except the completed certificate request objects are converted.

- For the ICL, all certificate objects except the expired certificates for which the user provided the public key, are converted.

Note: If there are numerous completed certificate requests and the expired certificates that are created with user-supplied keys, it can take longer to run the utility with the `-a` option.

Examples

This sample script converts the version of the record, depending on the source version specified. A record is either converted from `version 0` to `version 1` format, or from `version 1` format to `version 0` format. The names of the data sets are as follows:

pkisrvd.vsam.ost
Specifies the source object store VSAM data set.

pkisrvd.vsam.icl
Specifies the source ICL VSAM data set.

pkisrvd.vsam.newost
Specifies the target object store VSAM data set.

pkisrvd.vsam.newicl
Specifies the target ICL VSAM data set.

All of the records in the VSAM data sets are converted, including those for completed requests and expired certificates.

```
vsamconv -s ost=\'pkisrvd.vsam.ost\', icl=\'pkisrvd.vsam.icl\'
-t ost=\'pkisrvd.vsam.newost\', icl=\'pkisrvd.vsam.newicl\'
```

Output

As the `vsamconv` utility runs, it reports its progress. For every 1000 records that are converted, the utility reports the total number of records converted.

If an error occurs, `vsamconv` displays a count of processed records and a count of converted records.

If the source and target version are the same, an error is issued.

Chapter 21. Using the certificate management protocol (CMP) with PKI Services

Certificate management protocol (CMP) is an internet protocol used to manage X.509 digital certificates within a PKI. It is described in RFC 4210 (tools.ietf.org/html/rfc4210) and uses the certificate request message format (CRMF) described in RFC 4211 (tools.ietf.org/html/rfc4211). A certificate request message object is used within the protocol to convey a request for a certificate to a certificate authority. CMP messages are ASN.1-encoded. PKI Services allows a CMP client to communicate with it to request, revoke, suspend and resume certificates.

Restrictions: The following restrictions apply to the PKI Services support for CMP:

1. PKI Services supports only a subset of the CMP messages, and only some fields in those messages. See "Support for CMP messages" on page 454 for a description of the support.
2. PKI Services supports only the HTTP protocol for CMP messages.

PKI Services implements CMP through a CGI program. The tcp-message is sent to the PKI CMP CGI program by HTTPS POST, as specified in *Internet X.509 Public Key Infrastructure -- Transport Protocols for CMP* . The entire POST body is the message and the mime-type for both requester and responder (client and server) is `application/pkixcmp`.

Note: The `application/pkixcmp` mime-type requires that the entire tcp-message be Base64-encoded.

When a CMP client sends a request to the HTTP Server, it must send the request directly to the HTTP Server (and port number) that handles the client authentication requests. The request cannot be handled by a redirect statement. The shipped sample `vhost1443.conf` is for client authentication. The following two entries are specifically set up needed for the CMP client:

```
ScriptAliasMatch ^/(PKIServ|Customers)/(clientauth-cgi|clientauth-cgi-bin)/(.*)
"<application-root>/PKIServ/clientauth-cgi-bin/$3"

<LocationMatch "^/(PKIServ|Customers)/clientauth-cgi-bin/auth/pkicmp">
CharsetOptions NoTranslateRequestBodies
</LocationMatch>
```

Table 77 on page 453 shows the format of version 10 tcp-messages (the only existing version):

Table 77. Format of tcp-messages

Field	Length	Contents
Length	32 bits	The length of the rest of the tcp-message (the length of the CMP message + 3)
Version	8 bits	10
Flags	8 bits	The least significant bit indicates a closed connection. The other bits are unused.

Table 77. Format of tcp-messages (continued)

Field	**Length**	**Contents**
Message type	8 bits	Supported types are: • `pkiReq`, value X'00', indicating a synchronous request • `pkiRep`, value X'05', indicating a synchronous response • `errorMsgRep`, value X'06', indicating an error
Value	Variable, based on the length of the CMP message	The ASN.1 encoded CMP message

The communication between the CMP client and the CGI program is over HTTPS only. Client authentication is required. The client (the CMP requester) needs to have a certificate installed in RACF under the client's ID. This certificate is used by the requester to authenticate itself, and its owner ID is used to access the PKI Services functions.

Support for CMP messages

PKI Services supports the following request message types from the client:

- Certificate request message (type `cr`)
- Revocation request message (type `rr`)
- PKCS #10 certificate request message (type `p10cr`)

and responds with one of the following response messages:

- Certificate response message (type `cp`)
- Revocation response message (type `rp`)
- Error message (type `error`)

Each message supported by PKI Services contains the following parts:

- The header, containing information common to many messages
- The body, containing information specific to the message
- Optionally, certificates that might be useful to the recipient

Table 78 on page 454 identifies the fields in the `PKIMessage` structure defined in RFC 4210 (tools.ietf.org/html/rfc4210) that PKI Services supports.

Table 78. Supported fields in the `PKIMessage` structure

Field name	**Notes**
`header`	See Table 79 on page 455.
`body`	See Table 80 on page 455.

Table 78. Supported fields in the PKIMessage structure (continued)	
Field name	**Notes**
extraCerts	This field can be used by the client on a certificate request message (cr) when PKI Services is generating the public and private key for the requested certificate. This field can contain a list of x.509 certificates to be used as recipients of the private key to be returned by PKI Services. If this field is present, the _PKISERV_CMP_HONOR_CLIENT_CERTS environment variable determines whether extra certificates are allowed, and how many are allowed. If allowed, and if PKI Services generates the public and private key pair for the request, each certificate has a recipientInfo structure added to the returned encrypted private key (PKCS #7 EnvelopedData structure). For information about the _PKISERV_CMP_HONOR_CLIENT_CERTS environment variable, see Table 89 on page 471.

Table 79 on page 455 identifies the fields in the `PKIHeader` structure defined in RFC 4210 (tools.ietf.org/html/rfc4210) that PKI Services supports.

Table 79. Supported fields in the PKIHeader structure

Field name	Notes
`pvno`	
`sender`	
`recipient`	If this `GeneralName` field is in the form of a `directoryName` in a `cr`, `p10cr`, or `rr` message, it can be used to determine the PKI Services CA domain to which the request is directed. For information about how PKI Services determines the CA domain, see "Determining the CA domain to which a request is routed" on page 460.
`transactionID`	
`generalInfo`	The only `InfoTypeAndValue` recognized by PKI Services is `ImplicitConfirm`, which is required for `cr` and `p10cr` messages. This field is ignored if present on an `rr` request message.

Table 80 on page 455 identifies the values in the PKIBody structure defined in RFC 4210 (tools.ietf.org/html/rfc4210) that PKI Services supports. These are the CMP message types.

Table 80. Supported values in the PKIBody structure. These are the CMP message types that PKI Services supports.

Value	Description	Notes
`cr`	Certificate request	See Table 81 on page 456.
`cp`	Certificate response	See Table 83 on page 458.
`p10cr`	PKCS #10 certificate request	See Table 82 on page 457.
`rr`	Revocation request	See Table 84 on page 459.

Table 80. Supported values in the PKIBody structure. These are the CMP message types that PKI Services supports. (continued)

Value	Description	Notes
rp	Revocation response	See Table 85 on page 459.
error	Error message	See Table 86 on page 460.

Support for the CMP certificate request message (type `cr`)

Table 81 on page 456 identifies the fields that PKI Services supports in the data structure defined in RFC 4211 for the CMP certificate request message (type `cr`).

Table 81. Supported fields in the CMP certificate request message (type `cr`)

	Field name	Notes
In the `CertReqMsg` structure:		PKI Services supports a single `CertReqMsg` in the `CertReqMessages` field, and rejects a `cr` message with more than one `CertReqMsg`.
	`certReq`	
	`popo`	
In the `ProofOfPossession` structure:		
	`signature`	`signature` is the only supported choice. It should only be present if the CMP client has supplied `publicKey` in the `CertTemplate` structure. The `POPOSigningKey` structure must not contain a `poposkInput` field.
In the `CertRequest` structure:		
	`certReqId`	
	`certTemplate`	
In the `CertTemplate` structure:		
	`version`	
	`serialNumber`	
	`signingAlg`	
	`issuer`	If supplied, this field is used with the _PKISERV_CMP_DOMAIN_ISSUER*n* environment variables to determine to which PKI Services CA domain to route the request. For information about the _PKISERV_CMP_DOMAIN_ISSUER*n* environment variables, see Table 87 on page 466. For information about how PKI Services determines the CA domain, see "Determining the CA domain to which a request is routed" on page 460.

Table 81. Supported fields in the CMP certificate request message (type `cr`) (continued)

	Field name	**Notes**
	`validity`	If supplied, the _PKISERV_CMP_HONOR_CLIENT_DATES environment variable must set to **1**; otherwise the `cr` message is rejected. For information about the _PKISERV_CMP_HONOR_CLIENT_DATES environment variable, see Table 88 on page 467.
	`subject`	If omitted, the `cr` message is rejected.
	`publicKey`	Optional; if omitted PKI Services generates the public and private keys for the certificate request using environment variables to determine the key type and size.
	`extensions`	If the _PKI_CMP_HONOR_CLIENT_EXTS environment variable is not set to **1** and `extensions` is specified, the message is rejected. If the environment variable is set to **1**, `extensions` is honored if present, but is not required.

Support for the CMP PKCS #10 certificate request message (type `p10cr`)

Table 82 on page 457 identifies the fields that PKI Services supports in the data structure defined in RFC 4210 (tools.ietf.org/html/rfc4210) for the CMP PKCS #10 certificate request message (type `p10cr`)).

Table 82. Supported fields in the CMP PKCS #10 certificate request message (type `p10cr`)

	Field name	**Notes**
In the `CertificationRequest` structure:		When this message type is received by PKI Services, the recipient field is used with the _PKISERV_CMP_DOMAIN_ISSUER*n* environment variables to determine to which PKI Services CA domain to route the request, provided the recipient is encoded as a `directoryName`. For information about the _PKISERV_CMP_DOMAIN_ISSUER*n* environment variables, see Table 87 on page 466.
	`certificationRequestInfo`	
	`signatureAlgorithm`	
	`signature`	
In the `CertificationRequestInfo` structure:		
	`version`	
	`subject`	
	`subjectPKInfo`	

Table 82. Supported fields in the CMP PKCS #10 certificate request message (type `p10cr`) (continued)

	Field name	Notes
	`attributes`	If the _PKI_CMP_HONOR_CLIENT_EXTS environment variable is not set to **1** and the `extensionReq` attribute is specified, the message is rejected. If the environment variable is set to **1,** the `reqExtensions` attribute is honored if present, but is not required.

Support for the CMP certificate response message (type cp)

Table 83 on page 458 identifies the fields that PKI Services supports in the data structure defined in RFC 4210 (tools.ietf.org/html/rfc4210) for the CMP certificate response message (type `cp`). The `cp` message is returned to the CMP client for a successful `cr` or `p10cr` request.

Table 83. Support for fields in the CMP certificate response message (type `cp`)

	Field name	Notes
In the `CertRepMessage` structure:		`privateKey` is only returned in response to a `cr` message that does not specify `publicKey`.
	`caPubs`	PKI Services does not use this element.
	`response`	
In the `CertResponse` structure:		PKI Services returns only one `CertResponse` for a certificate request.
	`certReqId`	
	`status`	
	`certifiedKeyPair`	
	`rspInfo`	PKI Services does not use this element.
In the `CertifiedKeyPair` structure:		
	`certOrEncCert`	
	`privateKey`	PKI Services sets the object ID for `intendedAlg` to the PKCS #7 OID, which is 1.2.840.113549.1.7. `encValue` is a bit string encapsulation of a PKCS #7 `EnvelopedData` structure whose encrypted content is the DER-encoded private key for the issued certificate.

Support for the CMP revocation request message (type `rr`)

Table 84 on page 459 identifies the fields that PKI Services supports in the data structure defined in RFC 4210 (tools.ietf.org/html/rfc4210) for the CMP revocation request message (type`rr`).

Table 84. Supported fields in the CMP revocation request message (type `rr`)

	Field name	**Notes**
In the `RevDetails` structure:		PKI Services supports a single `RevDetails` in the `RevReqContent` message, and rejects an `rr` message with more than one `RevDetails` sequence.
	`certDetails`	PKI Services requires that `certDetails` contains the serial number of the certificate to be revoked, suspended, or resumed.
	`crlEntryDetails`	PKI Services recognizes only the `cRLReason` extension and ignores all other non-critical extensions in `crlEntryDetails`. If an extension other than the `cRLReason` extension is present and marked critical, PKI Services rejects the `rr` message.

Support for the CMP revocation response message (type `rp`)

Table 85 on page 459 identifies the fields that PKI Services supports in the data structure defined in RFC 4210 (tools.ietf.org/html/rfc4210) for the CMP revocation response message (type`rp`). PKI Services returns the `rp` message to the CMP client for a successful `rr` request.

If PKI Services receives a type `rp` message from a CMP client, it returns an `errorMessage` response.

Table 85. Supported fields in the CMP revocation response message (type `rp`)

	Field name	**Notes**
In the `RevRepContent` structure:		
	`status`	
	`revCerts`	PKI Services returns `revCerts` in the `rp` message when the corresponding `rr` message contains an `issuer` in `certDetails` or specifies a recipient in the form of a `directoryName`.
In the `PKIStatusInfo` structure:		
	`status`	
	`statusString`	
	`failInfo`	

Support for the CMP error message (type `error`)

Table 86 on page 460 identifies the fields that PKI Services supports in the data structure defined in RFC 4210 (tools.ietf.org/html/rfc4210) for the CMP revocation request message (type `error`). The error codes and details that can be returned are described in "Messages and codes returned from the CMP functions" on page 473.

If PKI Services receives a type `error` message from a CMP client, it returns an `errorMessage` response.

Table 86. Supported fields in the CMP error message (type `error`)

	Field name	Notes
In the `ErrorMsgContent` structure:		
	`pKIStatusInfo`	
	`errorCode`	This field is present in all type `error` messages that PKI Services returns. For the error codes and corresponding details that can be returned, see "Messages and codes returned from the CMP functions" on page 473.
	`errorDetails`	

Determining the CA domain to which a request is routed

To determine the CA domain to which it routes a request, the CMP CGI program first tries to determine the issuer distinguished name:

- If the message type is `cr` (certificate request), the `issuer` field of the `CertTemplate` structure is used as the issuer distinguished name if it is present. If it is not, the `recipient` field in the message header is used as the issuer distinguished name if it is in the form of a Directory Name (distinguished name). If the `recipient` field is not in the form of a Directory Name, an issuer distinguished name is not used to determine the CA domain name; instead, the URL to which the CMP request was sent is used to determine the CA domain.
- If the message type is `p10cr` (PKCS #10 certificate request message), the `recipient` field in the message header is used as the issuer distinguished name if it is in the form of a Directory Name (distinguished name). If the recipient field is not in the form of a Directory Name, an issuer distinguished name is not used to determine the CA domain name; instead, the URL to which the CMP request was sent is used to determine the CA domain.
- If the message type is `rr` (revoke request), the issuer and serial number fields of the `CertDetails` field are used as the issuer distinguished name and certificate serial number to be revoked or suspended. If the serial number is not present, the request is rejected. If the issuer field is not present, the `recipient` field in the message header is used as the issuer distinguished name if it is in the form of a Directory Name (distinguished name). If the `recipient` field is not in the form of a Directory Name, an issuer distinguished name is not used to determine the CA domain name; instead, the URL to which the CMP request was sent is used to determine the CA domain.

If the CMP CGI program was able to determine the issuer distinguished name, and the request is a certificate request (type `cr` or `p10cr`), the CMP CGI program does the following processing to determine to which CA domain it routes the request:

1. The CMP CGI program compares the issuer distinguished name that is extracted from the request in string format to the values defined in the _PKISERV_CMP_DOMAIN_ISSUER*i* environment variables (where *i* is 1 through the number of CA domains). The comparison is made by comparing the relative distinguished names in order of specification (first from most specific to least specific, then least specific to most specific). For example, if the issuer distinguished name in a request message is OU=STG,O=IBM,C=US, it would match a _PKISERV_CMP_DOMAIN_ISSUER*i* environment variable whose value was set to either of the following values:
   - OU=STG,O=IBM,C=US (most specific first)
   - c=us,o=ibm,ou=stg (least specific first)

The comparison is made without regard to the character case (case-insensitive). Some differences in spacing are allowed. For example, "O=IBM" matches "O = IBM", but not "O=I B M"

If a match is found for the issuer distinguished name, the CMP CGI continues to step "2" on page 461. If no match is found, it uses the URL to which the CMP request was sent to determine the CA domain.

2. The number (*i*) of the matching _PKISERV_CMP_DOMAIN_ISSUER*i* environment variable is used to read the domain name environment variable _PKISERV_CMP_DOMAIN_NAME*i*_*n*. The _PKISERV_CMP_DOMAIN_NAME*i*_*n* with the highest value of *n* is used because it represents the current domain name for the CA for accepting new certificate requests. (When the value of *n* is greater than 1, at least one CA rollover occurred.) For example, if the issuer distinguished name in a `cr` message matched the value in _PKISERV_CMP_DOMAIN_ISSUER3, and one CA rollover occurred for that CA, the CA domain name is retrieved from the _PKISERV_CMP_DOMAIN_NAME3_2 environment variable. If the _PKISERV_CMP_DOMAIN_NAME*i*_*n* environment variable cannot be read, the CMP request is rejected.

If the CMP CGI program was able to determine the issuer distinguished name, and the request is a revoke request (type `rr`), and a serial number was present in the request, the CMP CGI program does the following processing to determine to which CA domain it routes the request:

1. The CMP CGI program compares the issuer distinguished name that is extracted from the request in string format to the values defined in the _PKISERV_CMP_DOMAIN_ISSUER*i* environment variables (where *i* is 1 through the number of CA domains). The comparison is made in the same manner as described in step "1" on page 460 for `cr` and `p10cr` requests. If a match is found for the issuer distinguished name, the CMP CGI continues to step "2" on page 461. If no match is found, it uses the URL to which the CMP request was sent to determine the CA domain.
2. The number (*i*) of the matching _PKISERV_CMP_DOMAIN_ISSUER*i* environment variable is then used to read the serial number domain name environment variables _PKISERV_CMP_DOMAIN_FSTSN*i*_*n*. Each _PKISERV_CMP_DOMAIN_FSTSN*i*_*n* environment variable is read, starting with an *n* value of 1, and compared to the serial number retrieved from the `rr` request.

   - If there are no _PKISERV_CMP_DOMAIN_FSTSN*i*_*n* environment variables that are defined, the domain name is read from the _PKISERV_CMP_DOMAIN_NAME*i*_1 environment variable.
   - If the serial number is greater than the environment variable value and less than the next environment variable value (or is the last environment variable), the domain name is read from the corresponding _PKISERV_CMP_DOMAIN_NAME*i*_*n* environment variable.

     Example: An `rr` request is made by a CMP client specifying an issuer distinguished name of OU=STG, O=IBM, C=US, and the following sample is an excerpt from the defined environment variables:

     ```
     SetEnv _PKISERV_CMP_DOMAIN_ISSUER3 OU=STG,O=IBM,C=US
     SetEnv _PKISERV_CMP_DOMAIN_NAME3_1 STG_CA
     SetEnv _PKISERV_CMP_DOMAIN_NAME3_2 STG_CA2
     SetEnv _PKISERV_CMP_DOMAIN_NAME3_3 STG_CA3
     SetEnv _PKISERV_CMP_DOMAIN_FSTSN3_1 3
     SetEnv _PKISERV_CMP_DOMAIN_FSTSN3_2 12500
     SetEnv _PKISERV_CMP_DOMAIN_FSTSN3_3 25000
     ```

     If the serial number that is specified in the `rr` request is 20000 decimal, the revocation request is routed to the STG_CA2 domain because STG_CA2's first serial number is less than 20000, and the first serial number that is issued by STG_CA3 is greater than 20000.

How PKI Services interprets distinguished names (DNs) on CMP requests

The subject distinguished name is encoded in a CMP certificate request from the CMP client. The order in which the relative distinguished names (RDNs) are placed in the subject field by the client is the order in which the RDNs appear in the issued certificate. PKI Services interprets the order to be least significant RDN first and most significant RDN last. This becomes important when PKI Services posts the issued certificate to an LDAP server, because the LDAP server defines a root suffix that it allows objects and attributes to be stored under. If the LDAP server has a defined root suffix of "C=US", and PKI Services attempts to post a certificate with a subject name "CN=Gumby,O=IBM,C=US", the request succeeds because the string format of the subject name has the C=US as the rightmost RDN, and that is the defined

LDAP root suffix. If however, the CMP client encoded the subject name in the reverse order, the subject name string that PKI uses to post the certificate would be "C=US,O=IBM,CN=Gumby". This post request fails because the interpreted root suffix of CN=Gumby would not exist in LDAP. **Guideline:** When encoding a subject distinguished name in a certificate request, clients should place the least significant RDN first and the most significant RDN last.

Example: This example shows the encoded form of `CN=Gumby,O=IBM,C=US`:

```
SEQUENCE {
. SET {
. . SEQUENCE {
. . . OBJECT IDENTIFIER countryName (2 5 4 6)
. . . PrintableString 'US'
. . . }
. . }
. SET {
. . SEQUENCE {
. . . OBJECT IDENTIFIER organizationName (2 5 4 10)
. . . PrintableString 'IBM'
. . . }
. . }
. SET {
. . SEQUENCE {
. . . OBJECT IDENTIFIER commonName (2 5 4 3)
. . . PrintableString 'Gumby'
. . . }
. . }
. }
```

When the PKI Services CMP CGI program receives a CMP request from a CMP client, it attempts to determine the target CA domain to route the request to using the supplied issuer DN or the recipient (if it is in the form of a DN). The CMP CGI program interprets the encoded DN values to be the least significant RDN first and most significant RDN last. The program builds a string representation of the issuer or recipient DN from left to right starting with the last RDN in the sequence and ending with the first RDN. The CMP CGI program then compares the string that it built to the values of the _PKISERV_CMP_DOMAIN_ISSUER*x* environment variables. During the comparison of the strings, the program tries the comparison in both ways, both right to left and left to right, so if the distinguished name in the request is OU=Master CA,O=IBM,C=US it matches a _PKISERV_CMP_DOMAIN_ISSUER*x* value of either OU=Master CA,O=IBM,C=US, or C=US,O=IBM,OU=Master CA.

Setting up a client to make CMP requests to PKI Services

Before a client (the CMP requester) can make requests to PKI Services through CMP, the client must have a certificate installed in the RACF database under the client's RACF user ID. The certificate is used by the requester to authenticate the client, and the client's user ID is used to access the PKI Services functions. It must be signed by a CA certificate that is connected to the HTTP Server's key ring.

There are several ways to set up the client certificate. For example, you can have it generated elsewhere (not by PKI Services) and add it to the RACF database using the RACDCERT ADD command. Alternatively, you can generate the certificate yourself and add it to the RACF database using RACDCERT commands. The following instructions illustrate the latter approach.

Steps for setting up a certificate for a CMP requester

Perform the following steps to set up a certificate in the RACF database for a CMP requester.

Before you begin

You need to have RACF SPECIAL authorization, or authorization to the RACDCERT commands shown. For more information, see *z/OS Security Server RACF Command Language Reference*.

Procedure

1. Generate a certificate for the CMP requester in the RACF database, signed by a certificate that is in the HTTP Server's key ring.

Example:

```
RACDCERT ID(User123) GENCERT
SUBJECT(CN('Messager') OU('OrgUnitA') O('OrgA') C('AU'))
WITHLABEL('client') SIGNWITH(CERTAUTH LABEL('Local PKI CA'))
```

____

2. Export the certificate and its private key to a data set.

   Example:

   ```
   RACDCERT ID(User123) EXPORT(LABEL('client'))
   DSN('User123.private.eecert') FORMAT(PKCS12B64)
   PASSWORD('secret')
   ```

   Tip: You could use `FORMAT(PKCS12DER)` if you do not plan to use copy and paste operations to transport the certificate and private key to the client system.

   ____

3. Transport the certificate to the system where the CMP client will run. You can do this using FTP, or perhaps by copy and paste operations, depending on the CMP client software.

   ____

Results

When you are done, the client can make CMP requests to PKI Services.

Setting up PKI Services to process CMP requests

Before PKI Services can process CMP requests, you must do the following tasks:

1. Enable the CMP support by setting the `EnableCMP` configuration parameter.
2. If you want PKI Services to create private keys for CMP clients, you must set up ICSF and set up PKI Services to encrypt the private keys.

Enabling the CMP support

The PKI Services configuration file, `pkiserv.conf`, contains a parameter in the **CertPolicy** section that enables the CMP support. By default this parameter, `EnableCMP`, is set to false. If the parameter is set to false or omitted from the configuration file, PKI Services rejects all CMP requests.

To enable the CMP support, find the following lines in the configuration file:

```
# Enable the Certificate Management Protocol (CMP)
# T = True, CMP is enabled
# F = False, CMP is disabled (default if not specified)
#
EnableCMP=F
```

Change `EnableCMP=F` to `EnableCMP=T`, and restart PKI Services.

Setting up PKI Services to create private keys for CMP clients

PKI Services can create private keys for CMP clients and return a private key with a certificate. It uses the PKCS #11 API provided by ICSF to create private keys. Note, however, that PKI Services does not archive the private keys in the ICSF token data set (TKDS), as it does for private keys that it creates for certificate requests it receives from the end-user web application. To allow PKI Services to create private keys, you must ensure that the ICSF programmer has installed and configured ICSF, and has set up the TKDS. For more information, see "Installing and configuring ICSF" on page 33.

Note: You do not need to perform any of the other tasks that are described in "Steps for setting up PKI Services to generate keys for certificate requests" on page 326, such as setting the `TokenName` parameter in the configuration file, to allow the PKI Services CMP CGI program to generate private keys for CMP clients. Those tasks apply only to private key generation done by the PKI Services daemon, for certificates requested from the PKI Services web application.

Determining the source of certificates used to encrypt the returned private key

If a CMP client requests a certificate for which PKI Services creates the keys, PKI Services must encrypt the private key before returning it with the certificate to the recipient. There are three ways that the encryption of the private key can be done, and the method that is used is determined in the following order:

1. If the client provides an `extraCerts` field in the `PKIMessage` structure and the number of certificates in the `extraCerts` field does not exceed the value of the _PKISERV_CMP_HONOR_CLIENT_CERTS_*domain* environment variable, the CMP CGI program encrypts the private key in a manner in which it can be decrypted by each of the certificates in the `extraCerts` field.
2. If the `extraCerts` field is not specified in the request message, and the _PKISERV_CMP_KEYRING_*domain* environment variable is set, the private key is encrypted in a manner in which it can be decrypted by each of the certificates in the specified key ring.
3. If the `extraCerts` field is not specified in the request message, and no key ring is defined, the private key is encrypted with the public key of the certificate used to establish the secure client authentication session to the CMP CGI program.

Steps for setting up PKI Services to encrypt returned private keys with certificates in a key ring

For PKI Services to encrypt private keys using certificates in a key ring, you must set up a key ring containing a digital certificate for each recipient.

Before you begin

- You need to have RACF SPECIAL authorization, or sufficient authority to the following resources in the FACILITY class:
  - IRR.DIGTCERT.ADDRING
  - IRR.DIGTCERT.ADD
  - IRR.DIGTCERT.CONNECT
- You must have a certificate for each certificate recipient, in a data set.

Note: This example assumes that you are using certificates that were created somewhere else. Alternatively, you could create the certificates using the RACF command RACDCERT GENCERT. If you take this approach, you need authorization to the resource IRR.DIGTCERT.GENCERT in the FACILITY class.

Procedure

1. Set the HTTP Server environment variable _PKISERV_CMP_KEYRING:

   ```
   _PKISERV_CMP_KEYRING_domain RACF_userID/ring_name
   ```

   RACF_userID can be any RACF user ID; for example, the PKI Services daemon user ID, or the CMP requester user ID. *ring_name* is a name that you choose, and use when you create the key ring.

   ____

2. Create a RACF key ring, specifying the RACF user ID and the ring name that you specified in step "1" on page 464.

   ```
   RACDCERT ID(RACF_userID) ADDRING(ring_name)
   ```

   ____

3. Add certificates for each recipient to the RACF database, using the RACF user ID that you specified in step "1" on page 464.

```
RACDCERT ID(RACF_userID) ADD(dataset_1) WITHLABEL('label_1')TRUST
RACDCERT ID(RACF_userID) ADD(dataset_2) WITHLABEL('label_2')TRUST⋮
RACDCERT ID(RACF_userID) ADD(dataset_n) WITHLABEL('label_n')TRUST
```

dataset_n is the name of the data set containing the certificate for recipient *n*. *label_n* is the label to be associated with the certificate for recipient *n*.

____

4. Add the digital certificates to the key ring that you created in step "2" on page 464.

   ```
   RACDCERT ID(RACF_userID) CONNECT(LABEL('label_1') RING(ring_name))
   RACDCERT ID(RACF_userID) CONNECT(LABEL('label_2') RING(ring_name))
   ⋮
   RACDCERT ID(RACF_userID) CONNECT(LABEL('label_n') RING(ring_name))
   ```

   label_n is the label you associated with the certificate for recipient *n* in step "3" on page 464.

   ____

5. Authorize the PKI Services CMP CGI program to access the key ring. This program runs with the RACF user ID that the client-supplied certificate maps to, so you must give that RACF user ID access to the key ring. You can use one of two methods:

   - (Preferred) Define a profile in the RDATALIB class for the key ring and give each CMP client user ID READ access:

     ```
     RDEFINE RDATALIB ring_owner.ring_name.LST UACC(NONE)
     PERMIT ring_owner.ring_name.LST CLASS(RDATALIB) ID(client_user_id)
     ACCESS(READ)
     SETROPTS RACLIST(RDATALIB) REFRESH
     ```

     ring_owner is the RACF user ID you that specified in step "2" on page 464, and *ring_name* is the ring name that you specified.

   - (Alternative) Define a profile in the FACILITY class for IRR.DIGTCERT.LISTRING. Give the ring owner READ access and the client user IDs UPDATE access:

     ```
     RDEFINE FACILITY IRR.DIGTCERT.LISTRING UACC(NONE)
     PERMIT IRR.DIGTCERT.LISTRING ID(ring_owner) ACCESS(READ)
     PERMIT IRR.DIGTCERT.LISTRING ID(client_user_id) ACCESS(UPDATE)
     ```

   ____

6. If either the DIGTCERT or DIGTRING class is RACLISTed, refresh the RACLISTed classes to activate your changes.

   ```
   SETROPTS RACLIST(DIGTCERT, DIGTRING) REFRESH
   ```

   ____

Results

When you are done, you have set up PKI Services to encrypt returned private keys with certificates in a RACF key ring.

Setting up the HTTP Server for CMP

You can pass information about CMP requests through HTTP Server environment variables.

- For IBM HTTP Server - Powered by Apache, the environment variables are specified using the `SetEnv` directive in the virtual host file for SSL requests with client authentication, which by default is `vhost1443.conf`.

For information about IBM HTTP Server - Powered by Apache environment variables, see the WebSphere Application Server Knowledge Center (www.ibm.com/support/knowledgecenter/SSEQTP). The variables that you can set are described in Table 87 on page 466, Table 88 on page 467, and Table 89 on page 471.

Table 87. HTTP Server environment variables used to determine the CA

Environment variable name	Description
_PKISERV_CMP_DOMAIN_ISSUER*x*	CA's subject distinguished name with comma-separated RDNs **Example:** `SetEnv _PKISERV_CMP_DOMAIN_ISSUER1 "CN=Issuer CA,OU=Lab,O=IBM,C=AU"`

_PKISERV_CMP_DOMAIN_NAME*x_y*	Issuer's PKI CA domain name. The domain name variable is limited to 8 characters beginning with an alphabetic character or an underscore "_" and no embedded spaces. For an unnamed domain, this variable must be set to `<NONE>`. A rollover of an unnamed domain requires the new domain to be named.
	Example: If there are, or have been, 3 domains of certificates for the same issuer, there would be 3 entries in the environment variables file:
	`SetEnv _PKISERV_CMP_DOMAIN_NAME1_1 CardCA1`
	`SetEnv _PKISERV_CMP_DOMAIN_NAME1_2 CardCA2`
	`SetEnv _PKISERV_CMP_DOMAIN_NAME1_3 CardCA3`
	Example: The domain is unnamed.
	`SetEnv _PKISERV_CMP_DOMAIN_NAME1_1 <NONE>`

_PKISERV_CMP_DOMAIN_FSTSN*x_y*	Starting serial number, in decimal, for the first certificate that is issued by _PKISERV_CMP_DOMAIN_NAME*x_y*. **Example:** `SetEnv _PKISERV_CMP_DOMAIN_FSTSN1_2 1001`

Notes:

1. *x* represents any of the available certificate issuers from 1 to the maximum number of issuers. If there are 2 issuers of certificates, there would be 2 entries in the configuration file. The values of *x* are consecutive integers starting with 1.
2. *y* represents any of the available CA domains that issue or have issued certificates on behalf of _PKISERV_CMP_DOMAIN_ISSUER*x*. The values of *y* are consecutive integers starting with 1.
3. Environment variable values must be enclosed in quotations if they include white space.
4. Any comma in a relative distinguished name (RDN) value must be escaped by placing a backslash immediately before the comma in the _PKISERV_CMP_DOMAIN_ISSUER*x* variable. For example, if your Organization RDN value is "Widgets, Inc.", the RDN must be specified as "O=Widgets\, Inc.". The commas that separate the RDNs must not be escaped.

   Example:

   ```
   CN=Issuer CA,OU=Lab,O=Widgets\, Inc.,C=AU
   ```

Table 88. HTTP Server environment variables used to control the content of the certificate within a CA

Environment variable name	**Description**

_PKISERV_CMP_KEYTYPE_*domain*	Specifies the key type or encryption algorithm to be used to generate the key pair. PKI Services supports RSA, NISTECC and BPECC.
	Examples:
	`SetEnv _PKISERV_CMP_KEYTYPE_CardCA1 RSA`
	`SetEnv _PKISERV_CMP_KEYTYPE BPECC`

_PKISERV_CMP_KEYSIZE_*domain*	A three- or four-digit number that specifies the length of the key.
	• For RSA, minimum size is 512 (1024 if _PKISERV_CMP_SECUREKEY is set), maximum size is 4096, and the default is 1024. The value must be a multiple of two (256, if _PKISERV_CMP_SECUREKEY is set).
	• For NISTECC, valid sizes are 192, 224, 256, 384 and 521, and the default is 192.
	• For BPECC, valid sizes are 160, 192, 224, 256, 320, 384, 512, and the default is 192.
	Examples:
	`SetEnv _PKISERV_CMP_KEYSIZE_CardCA1 2048`
	`SetEnv _PKISERV_CMP_KEYSIZE 512`

_PKISERV_CMP_SECUREKEY_*domain*	Specifies whether to generate secure keys in the TKDS. The value 1 specifies that secure keys are to be generated. The value 0 specifies that clear keys are to be generated.
	If this variable is not set, the profile protecting the CLEARKEY resource in the CRYPTOZ class determines whether the key generated is a secure key or a clear key. If the profile allows clear key generation, the generated key is a clear key. If the profile restricts clear key generation, the generated key is a secure key. For example, the following command creates a profile that prevents the generation of clear keys from any of the CMP clients:
	`RDEF CRYPTOZ CLEARKEY.SYSTOK-SESSION-ONLY`
	`UACC(NONE)`
	In this case, AES256 is used to envelop the private key.

_PKISERV_CMP_SECUREKEY_KEYENCALG_*domain*	Specifies the algorithm to envelop the private key. Valid values are AES256, AES128, and TDES. If not set, it defaults to AES256. This variable is ignored if _PKISERV_CMP_SECUREKEY_*domain* is not set.

Table 88. HTTP Server environment variables used to control the content of the certificate within a CA (continued)

Environment variable name	Description

_PKISERV_CMP_HONOR_CLIENT_DATES_*domain*	Specifies whether to honor client-specified certificate validity dates. Valid values are 0 and 1.
	
	The value 1 specifies that PKI Services uses start and end dates that are specified in the validity field of the certificate request. If validity dates are not specified, PKI Services uses the values in the environment variables _PKISERV_CMP_NOTBEFORE_*domain* and _PKISERV_CMP_NOTAFTER_*domain*. If either the start date or the end date is specified and the other is not specified, PKI Services uses the value set by the corresponding environment variable for the unspecified value.
	
	The value 0, or the absence of this variable, indicates that PKI Services is to use the values in the environment variables _PKISERV_CMP_NOTBEFORE_*domain* and _PKISERV_CMP_NOTAFTER_*domain*. If the client specifies either a start or end date in the validity field of a `cr` message when the value of this variable is 0 or absent, PKI Services rejects the request.
	
	Examples:
	
	`SetEnv`
	`_PKISERV_CMP_HONOR_CLIENT_DATES_CardCA1 1`
	`SetEnv _PKISERV_CMP_HONOR_CLIENT_DATES 0`

_PKISERV_CMP_NOTBEFORE_*domain*	Specifies the number of days from day of issue to when the certificate becomes valid. Valid values are 0 - 30 days. The default is 0, specifying that the certificate is valid from the start of the current day.
	
	Examples:
	
	`SetEnv _PKISERV_CMP_NOTBEFORE_CardCA1 7`
	`SetEnv _PKISERV_CMP_NOTBEFORE 5`

_PKISERV_CMP_NOTAFTER_*domain*	Specifies the number of days from today's date to when the certificate expires. Valid values are 1 - 9999. The default is 365. The value 1 specifies that the certificate is valid until the end of the current day.
	
	Examples:
	
	`SetEnv _PKISERV_CMP_NOTAFTER_CardCA1 1825`
	`SetEnv _PKISERV_CMP_NOTAFTER 1580`

Table 88. HTTP Server environment variables used to control the content of the certificate within a CA (continued)

Environment variable name	Description

_PKISERV_CMP_HONOR_CLIENT_EXTS_*domain*	Specifies whether to honor the client-specified extensions Subject Alternate Name, Keyusage and Extended keyusage. Valid values are 0 and 1.
	
	The value 1 specifies that PKI Services uses the extensions Subject Alternate Name, Keyusage and Extended keyusage from the request.
	
	The value 0, or the absence of this variable, specifies that PKI Services is to use the extensions Keyusage and Extended keyusage from the environment variables _PKISERV_CMP_KEYUSAGE_*domain* and _PKISERV_CMP_EXTKEYUSAGE_*domain*. If the client specifies the extensions field of a `cr` message or the attributes field of a `p10cr` message when the value of this variable is 0 or absent, PKI Services rejects the request.
	
	Examples:
	
	`SetEnv _PKISERV_CMP_HONOR_CLIENT_EXTS_CardCA1`
	`1`
	`SetEnv _PKISERV_CMP_HONOR_CLIENT_EXTS 0`

_PKISERV_CMP_KEYUSAGE_*domain*	Blank-separated character strings defining the key usage to be added to requested certificates. Valid values are: `handshake`, `dataencrypt`, `certsign`, `docsign`, `digitalsignature`, `digitalsig`, `nonrepudiation`, `keyencipherment`, `keyenciph`, `keyencrypt`, `dataencipherment`, `dataenciph`, `keyagreement`, `keyagree`, `keycertsign`, `crlsign`, `cmcaa`, `cmcra`, `cmcas`, `pkinitkdc`, and `pkinitclientauth`. The values are not case-sensitive. No default value.
	
	Examples:
	
	`SetEnv _PKISERV_CMP_KEYUSAGE_CardCA1`
	`        "digitalsig keyencrypt"`
	`SetEnv _PKISERV_CMP_KEYUSAGE handshake`

_PKISERV_CMP_EXTKEYUSAGE_*domain*	Blank-separated character strings defining the extended key usage to be added to requested certificates. Valid values are: `serverauth`, `clientauth`, `codesigning`, `emailprotection`, `timestamping`, `ocspsigning`, `mssmartcardlogon`, `cmcaa`, `cmcra`, `cmcas`, `pkinitkdc`, and `pkinitclientauth`. The values are not case-sensitive. No default value.
	
	Examples:
	
	`SetEnv _PKISERV_CMP_EXTKEYUSAGE_CardCA1`
	`clientauth`
	`SetEnv _PKISERV_CMP_EXTKEYUSAGE serverauth`

Table 88. HTTP Server environment variables used to control the content of the certificate within a CA (continued)

Environment variable name	Description

_PKISERV_CMP_AUTHINFOACC_*domain*	Deprecated. Use _PKISERV_CMP_AUTHINFOACC*n_domain* instead. Ignored if you specify _PKISERV_CMP_AUTHINFOACC*n_domain*.
	
	A comma-separated two-part string specifying information that is used to create the `AuthorityInfoAccess` extension. The two-part string identifies the `accessMethod` and `accessLocation`. The `accessMethod` is either `OCSP` or `IdentrusOCSP` (case-insensitive). The `accessLocation` is a URI in the form `URI=access-url` or `URL=access-url`. The `access-url` must be an HTTP protocol. No default value.
	
	Examples:
	
	`SetEnv _PKISERV_CMP_AUTHINFOACC_CardCA1 OCSP,`
	`   URL=http://www.widgets.com/CardCA1/public-`
	`cgi/caocsp`
	`SetEnv _PKISERV_CMP_AUTHINFOACC OCSP,`
	`   URI=http://www.widgets.com/PKIServ/public-`
	`cgi/caocsp`

_PKISERV_CMP_AUTHINFOACC*n_domain*	Specifies one or more `AuthorityInfoAccess` extensions in the form of a comma-separated two-part string. The two-part string identifies the `accessMethod` and `accessLocation`. The `accessMethod` is either `OCSP` or `IdentrusOCSP` (case-insensitive). The `accessLocation` is a URI in the form `URI=access-url` or `URL=access-url`. The `access-url` must be an HTTP protocol. No default value. There can be multiple entries, where *n* is 1 for the first `AuthorityInfoAccess` extension and increases sequentially for additional entries.
	
	Examples:
	
	`SetEnv _PKISERV_CMP_AUTHINFOACC1_CardCA1 OCSP,`
	`   URL=http://www.widgets.com/CardCA1/public-`
	`cgi/caocsp`
	`SetEnv _PKISERV_CMP_AUTHINFOACC2_CardCA1 OCSP,`
	`   URI=http://www.widgets.com/PKIServ/public-`
	`cgi/caocsp`

Table 88. HTTP Server environment variables used to control the content of the certificate within a CA (continued)

Environment variable name	Description

_PKISERV_CMP_CERTPOLICIES_*domain*	Specifies the certificate policies that are to be included in the issued certificates. The value is a set of numbers, which are separated by blanks, each representing one of the `PolicyName` values specified in the **CertPolicy** section of the PKI Services configuration file. Valid values are 1 - 99. No default value.
	Examples:
	`SetEnv _PKISERV_CMP_CERTPOLICIES_CardCA1 "1 2 3"`
	`SetEnv _PKISERV_CMP_CERTPOLICIES "1 4"`

_PKISERV_CMP_CUSTOMEXT*n*_*domain*	Specifies one or more custom extensions in the form of a comma-separated four-part string as indicated in the CustomExt field in the GENCERT CertPlist. There can be multiple entries, where *n* is 1 for the first custom extension and increases sequentially for additional entries. **Example:** `SetEnv _PKISERV_CMP_CUSTOMEXT1_CardCA1 1.3.6.1.4.1.311.20.2, N,BMP,myDomainController`

Notes:

1. _*domain* represents the domain name that is contained in the variable _PKISERV_CMP_DOMAIN_NAME*x*_*y*. For an unnamed domain, omit _*domain*.
2. Environment variable values must be enclosed in quotations if they include white space.

Table 89. HTTP Server environment variables used to configure the certificate recipients

Environment variable name	Description
_PKISERV_CMP_HONOR_CLIENT_CERTS_*domain*	Specifies the maximum number of extra input recipient certificates that can be supplied in the input message, by using the `extraCerts` construct in the PKIMessage structure. Valid values are 0 - 5. If omitted or set to a value of 0, certificate requests that contain `extraCerts` are rejected. **Example:** `SetEnv _PKISERV_CMP_HONOR_CLIENT_CERTS_CardCA1 3` For more information, see "Determining the source of certificates used to encrypt the returned private key" on page 464.

Table 89. HTTP Server environment variables used to configure the certificate recipients (continued)

Environment variable name	**Description**
_PKISERV_CMP_KEYRING_*domain*	Specifies the RACF user ID and the name of the key ring that is associated with that user ID that contains the certificates that are to be used to encrypt the private key for certificate request messages. **Example:** `SetEnv _PKISERV_CMP_KEYRING_CardCA1 CMPCLNT/CardKeyRing` For more information, see "Determining the source of certificates used to encrypt the returned private key" on page 464.

Table 90. HTTP Server environment variables used to control tracing

Environment variable name	**Description**
_PKISERV_CMP_TRACE	Specifies a bit mask enabling CMP trace options. No trace option is enabled if the bit mask is 0 and all trace options are enabled if the bit mask is 0xff. The bit mask can be specified as a decimal (*nnn*), octal (0*nnnn*) or hexadecimal (0x*hh*) value. These trace options are available: **0x01** CMP error messages **0x02** CMP informational messages **0x04** R_PKIServ callable service parameter list traces on entry and exit **0x08** Elapse time messages of events within the CMP program **0x10** CMP program function entry and exit trace messages **0x20** DER buffer display messages **0x40** Displays environment variables that are set at CMP program startup **Example:** `SetEnv _PKISERV_CMP_TRACE 0xff`

Table 90. HTTP Server environment variables used to control tracing (continued)

Environment variable name	Description
_PKISERV_CMP_TRACE_FILE	Specifies the name of the trace file. Defaults to `/tmp/pkicmp.%.trc`. The trace file is not used if the _PKISERV_CMP_TRACE environment variable is not defined or is set to 0. The current process identifier is included as part of the trace file name when the name contains a percent sign (%). For example, if _PKISERV_CMP_TRACE_FILE is set to `/tmp/pkicmp.%.trc` and the current process identifier is 247, the trace file name is `/tmp/pkicmp.247.trc`. **Guideline:** Because multiple copies of the CMP CGI program can run concurrently for multiple CMP clients, the value of _PKISERV_CMP_TRACE_FILE should include the percent sign (%) to prevent multiple copies of the CMP CGI program from writing to the same file. **Example:** `SetEnv _PKISERV_CMP_TRACE_FILE /tmp/pkicmp.%.trc`

Table 91. HTTP Server environment variable used to control the FIPS level

Environment variable name	Description
_PKISERV_CMP_FIPS_LEVEL	Specify the FIPS level the CMP program runs. **0** non FIPS mode (default) **1** FIPS 140-2 **2** SP800-131-A with exception **3** SP800-131-A without exception The value applies to all the domains. Make sure that the specified value agrees with the key size that is specified in the _PKISERV_CMP_KEYSIZE_ domain keyword. The value of `0` or, the absence of this variable, indicates that CMP is not running in FIPS mode. **Example:** `SetEnv _PKISERV_CMP_FIPS_LEVEL 1`

Tracing the PKI CMP CGI program

You can enable tracing of the PKI Services CMP CGI program using the environment variable _PKISERV_CMP_TRACE to set the trace options, and the environment variable _PKISERV_CMP_TRACE_FILE to specify the name of the trace file. A single trace file is created for each invocation of the CMP CGI program. For information about these environment variables, see Table 90 on page 472.

Messages and codes returned from the CMP functions

Most messages are returned to the caller as a CMP error response. In addition, all messages are echoed to the CGI program error log in the format:

```
date time pkicmp-> function: (error code) message text
```

For example:

```
Wed Oct 28 09:34:12 2009 pkicmp->processNotDays: (577) NotBefore date supplied is invalid(date is before
today).
```

Table 92. CMP error codes

Error code	**Explanation**
	Note: Error codes 06 - 99 are the reason codes from the RACF IRRSPX00 callable service.
06	Request queue serialization timeout occurred.
08	Request denied, not authorized.
12	An internal error has occurred during RACF processing.
20	Function code specified is not defined.
28	Certificate generation provider not available for specified CA domain
32	Incorrect value specified for CA domain.
40	Incorrect Reason Specified.
52	Parameter has an incorrect value.
56	Required field is missing from request.
60	Certificate generation provider error.
64	SerialNum has an incorrect length.
72	The status of the certificate has been changed by another process.
76	Conflicting fields names in CertPlist.
99	General error for other RACF callable service IRRSPX00 errors.
510	(510) Base64 decode of input message failed, error=*error-code*
511	(511) Base64 encode of output message failed, error=0x*hex-error code*
512	(512) Storage allocation failed (client certificate storage:*error-code*)
513	(513) Error occurred, HTTP access is forbidden.
514	(514) Error occurred, HTTP method was *HTTP method name* instead of POST.
515	(515) Failed to create CMP response message.
516	(516) Unsupported TCP Message protocol version.
517	(517) Unsupported TCP Message message type.
518	(518) Error occurred attempting to read the HTTP input message
519	(519) Unsupported CMP message type: *CMP message type specified*
520	(520) Key size (*envar name* envar) of *envar value* is not a multiple of {2 | 256}.
521	(521) Key size (*envar name* envar) of *envar value* is not valid for {RSA | secure RSA | NISTECC | BPECC} keys, must be {between 512-4096 | 1024-4096 | 192, 224, 256, 384, or 521 | 160, 192, 224, 256, 320, 384, or 512}.
522	(522) Storage allocation failed *element*:*size*

Table 92. CMP error codes (continued)

Error code	Explanation
523	(523) HonorClientDates (*envar name* envar) value of *envar value* is not numeric.
524	(524) HonorClientDates (*envar name* envar) value of *envar value* is not valid, expected 0 or 1.
525	(525) notBefore (*envar name* envar) value of *envar value* is not numeric.
526	(526) notAfter (*envar name* envar) value of *envar value* is not numeric.
527	(527) HonorClientCerts (*envar name* envar) value of *envar value* is not numeric.
528	(528) HonorClientCerts (*envar name* envar) value of *envar value* invalid, should be 0-5.
529	(529) HonorClientExts (*envar name* envar) value of *envar value* is not numeric.
530	(530) HonorClientExts (*envar name* envar) value of *envar value* invalid, should be 0 or 1.
531	(531) CMP Envar *envar name* value is not valid, expected {*valid values*}
533	(533) CMP Envar *envar name* missing in config file.
534	(534) CMP Envar *envar name* value of *envar value* is not numeric.
535	(535) CMP Envar *envar1 name* with value of *envar1 value* <= *envar2 name* with value of *envar2 value*.
536	(536) Key size (*envar name* envar)not specified, defaulting to *default key size value*.
537	(537) Key size (*envar name* envar) of *Key size value specified* is not numeric.
538	(538) KeyType (*envar name* envar) of *KeyType value specified* is not valid.
539	(539) *envar name* envar value length is greater than the maximum length of *maximum length*.
540	(540) Key size (*envar name* envar) of *value* is not compliant with FIPS level *value* specified
541	(541) Key type (*envar name* envar) of *value* is not compliant with FIPS level *value* specified
543	(543) request.extraCerts[*index value*].write() failed, status=*error code*
544	(544) gsk_decode_certificate failed, error code=0x*System SSL error code* - *System SSL brief error description*
545	(545) gsk_decode_base64 failed, error code=0x*System SSL error code* - *System SSL brief error description*
547	(547) gsk_open_keyring() failed: Error 0x*System SSL error code* - *System SSL brief error description*
548	(548) gsk_decode_certificate failed, error code=0x*System SSL error code* - *System SSL brief error description*
549	(549) gsk_get_record_by_index() failed: Error 0x*System SSL error code* - *System SSL brief error description*
550	(550) Specified Keyring *Keyring name* contains no certificates
552	(552) Failed to set FIPS state

Table 92. CMP error codes (continued)

Error code	**Explanation**
553	(553) gsk_decode_import_certificate failed, error code=0x*System SSL error code* - *System SSL brief error description*
554	(554) gsk_encode_private_key, error code=0x*System SSL error code* - *System SSL brief error description*
555	(555) gsk_make_enveloped_data_msg_extended failed, error code=0x*System SSL error code* - *System SSL brief error description*
556	(556) gsk_encode_export_certificate failed, error code=0x*System SSL error code* - *System SSL brief error description*
557	(557) gsk_construct_private_key_rsa failed, error code=0x*System SSL error code* - *System SSL brief error description*
558	(558) gsk_construct_public_key[ECC] failed, error code=0x*System SSL error code* - *System SSL brief error description*
559	(559) gsk_modify_pkcs11_key_label failed, error code=0x*System SSL error code* - *System SSL brief error description*
560	(560) gsk_make_enveloped_private_key_msg error failed, error code=0x*System SSL error code* -*System SSL brief error description*
562	(562) Triple Des Algorithm not available, using Single Des.
563	(563) Gencert succeeded, But no Transaction ID returned.
573	(573) Could not decode CMP message.
576	(576) Missing ImplicitConfirm in PKIHeader.
577	(577) {NotBefore | NotAfter} date supplied is invalid({cannot compute seconds since epoch | date is before today}).
581	(581) Validity supplied when not configured to honor client dates.
582	(582) No CA domain found for issuer *Issuer Distinguished name*.
584	(584) Number of extraCerts > HonorClientCerts (*envar name* envar) value of *envar value*.
587	(587) Critical crl extension oid=*Extension OID value* is not supported.
588	(588) crlReason extension value is not valid; decode error *error-code*.
589	(589) Serial number required in certTemplate.
590	(590) {Revoke/Suspend | Resume} of serial number *decimal-serial-number*(0x*hex-serial-number*) failed for CA Domain *Domain name*.
600	(600) Error encountered while encoding response body(*failing element*[:*error code*])
601	(601) Error encountered while encoding response header(*failing element*[:*error code*])
602	(602) {Attributes | Extensions} supplied when not configured to honor client extensions.
603	(603) Base64 encode of CertificationRequest failed, error=*error-code*.
604	(604) CertReqMsg with publicKey has missing or unsupported ProofOfPossesion

Table 92. CMP error codes (continued)

Error code	**Explanation**
606	(606) Error encountered while encoding CertReqMsg(*failing element*[:*error code*]).
607	(607) Error obtaining the current time of day(*failing step:error code*).
608	(608) Error retrieving information from the CMP request (*failing element*[:*error code*])
609	(609) {cr | rr} message does not contain only one {CertReqMsg | RevDetails}.
610	(610) Subject name absent from CertTemplate for a cr message
611	(611) Unsupported CMP message version (*version specified* not equal 2)
612	(612) Error initializing PKI Services configuration file (*configuration-file-name*)
613	(613) CA domain *domain-name* does not have CMP support enabled
614	(614) Error retrieving CMP environment variables
620	(620) Key type {[null] | *specified KEYTYPE value*} is not valid.
621	(621) Cannot initialize ICSF PKCS#11 interfaces (C_Initialize return code 0x*hex-return-code*)
622	(622) Error encountered while destroying a {Publik | Private} key object (return code 0x*hex-return-code*)
623	(623) Internal PKCS#11 API failure (*PKCS #11 API*) return code 0x*hex-return-code*
624	(624) {RSA | ECC} key generation failure (C_GenerateKeyPair return code 0x*hex-return-code*)
464453637	VSAM contention caused the request to fail. Retry the request.
464453634	VSAM contention caused the request to fail. Retry the request.

Part 5. Administering security for PKI Services

This part explains how to administer security for PKI Services.

- Chapter 22, “RACF administration for PKI Services,” on page 481 describes how to use RACF to administer security for PKI Services.

  The following tasks are covered:

  - “Authorizing users for the PKI Services administration group” on page 481
  - “Authorizing users for inquiry access” on page 481
  - “Administering HostIdMappings extensions” on page 482
  - “Locating your PKI Services certificates and key ring” on page 484
  - “Establishing PKI Services as an intermediate CA” on page 486
  - “Renewing your PKI Services CA and RA certificates” on page 488
  - “Recovering a CA certificate profile” on page 490
  - “Retiring and replacing the PKI Services CA private key” on page 491
  - “R_PKIServ (IRRSPX00 and IRRSPX64) callable service” on page 495
  - “Using encrypted passwords for LDAP servers” on page 501.

Chapter 22. RACF administration for PKI Services

This topic describes the tasks that the RACF administrator performs after PKI Services has been set up and customized.

The following tasks are covered:

- "Authorizing users for the PKI Services administration group" on page 481
- "Authorizing users for inquiry access" on page 481
- "Administering HostIdMappings extensions" on page 482
- "Locating your PKI Services certificates and key ring" on page 484
- "Establishing PKI Services as an intermediate CA" on page 486
- "Renewing your PKI Services CA and RA certificates" on page 488
- "Recovering a CA certificate profile" on page 490
- "Retiring and replacing the PKI Services CA private key" on page 491
- "R_PKIServ (IRRSPX00 and IRRSPX64) callable service" on page 495
- "Using encrypted passwords for LDAP servers" on page 501.

For more information about the RACF commands shown in this topic, see *z/OS Security Server RACF Command Language Reference*.

Authorizing users for the PKI Services administration group

You need to know how to add and delete members from the PKI Services administration group (by default, PKIGRP).

You might have set up multiple administration groups if you are using the PKISERV class to grant authorization on a granular level. For more information, see "Using the PKISERV class to control access to administrative functions" on page 499 and "Deciding the value of AdminGranularControl" on page 48.

Connecting members to the group

The PKI Services administration group is a RACF group containing the list of user IDs that are authorized to use PKI Services administration functions. To connect a member to the group, issue the following command, replacing *pkigroup_mem* with the member's user ID and *pkigroup* with the name of the PKI Services administration group (PKIGRP by default). (See Table 19 on page 55 for more information.)

```
CONNECT pkigroup_mem GROUP(pkigroup)
```

Note: You need to enter this command for each user ID in turn.

Deleting members from groups

To remove a user from a group, issue the following command, replacing *pkigroup_mem* with the user ID of the member you want to delete and *pkigroup* with the name of the PKI Services administration group (PKIGRP by default).

```
REMOVE pkigroup_mem GROUP(pkigroup)
```

Authorizing users for inquiry access

You can add groups of users who do not need the full administrative authority of users in the PKIGRP group. You can use the following procedure to authorize a new group for inquiry abilities, such as a help desk might require. The commands that are shown include variables whose names are appropriate for this scenario.

Steps for authorizing users for inquiry access

Before you begin

If you implemented the object store and ICL using VSAM data sets, you need to know the high-level VSAM data set qualifier that is used for the IKYSETUP variable *vsamhlq* value, in case your installation did not use the PKISRVD default. (See Table 19 on page 55.)

Procedure

Perform the following steps to add and administer a group that needs authority to query PKI Services information.

1. Add the new group.

   ```
   ADDGROUP HELPDESK OMVS(GID(197312))
   ```

2. Connect each member to the new group. Repeat for each user ID you need to connect.

   ```
   CONNECT OPER17 GROUP(HELPDESK)
   ```

3. Authorize the new group for READ access to the resources of PKI Services. Replace your installation's value for the data set's high-level qualifier if your installation did not use the PKISRVD default.

   ```
   PERMIT 'PKISRVD.**' ID(HELPDESK) ACCESS(READ)
   PERMIT IRR.RPKISERV.PKIADMIN CLASS(FACILITY)
      ID(HELPDESK) ACCESS(READ)
   SETROPTS GENERIC(DATASET) REFRESH
   SETROPTS RACLIST(FACILITY) REFRESH
   ```

   The SETROPTS commands activate the profiles that authorize READ access.

4. If necessary, you can remove a user from the group. The following example removes the user that you connected in Step “2” on page 482.

   ```
   REMOVE OPER17 GROUP(HELPDESK)
   ```

5. If necessary, you can delete the group. The following example deletes the group that you created in Step “1” on page 482.

   ```
   DELGROUP(HELPDESK)
   ```

Administering HostIdMappings extensions

You can add a `HostIdMappings` extension to certificates you create for certain users, allowing you to specify the user IDs that each user is able to use for login to particular servers (or hosts). Controlling an identity that is used for login purposes is an important security objective. Therefore, you must exercise administrative control in the following areas by authorizing:

- PKI Services as a highly trusted certificate authority whose certificates are honored when they contain `HostIdMappings` extensions
- Particular servers to accept logins from clients whose certificates contain `HostIdMappings` extensions

Steps for administering HostIdMappings extensions

Perform the following steps to allow the web server to accept logins from clients who have been issued PKI Services certificates with `HostIdMappings` extensions:

1. Determine if PKI Services is defined as a highly trusted certificate authority on your system by listing its certificate authority definition by using the RACDCERT CERTAUTH LIST command.

   Example:

   ```
   RACDCERT CERTAUTH LIST(LABEL('Local PKI CA'))
   ```

   Check the `Status` information near the beginning of the output listing for the `HIGHTRUST` attribute.

   ____

2. If not already defined, add the `HIGHTRUST` attribute to the certificate authority definition for PKI Services.

   Example:

   ```
   RACDCERT CERTAUTH ALTER(LABEL('Local PKI CA')) HIGHTRUST
   ```

   ____

3. Define a resource in the SERVAUTH class for each server (host) name you want your web server to honor when accepting logins for certificates containing `HostIdMappings` extensions. The resource name follows the format: `IRR.HOST.`*`hostname`*. The *hostname* is the value of the `HostIdMappings` extension entry pertaining to the z/OS host system you are administering (without the subject ID portion). This is usually a domain name, such as `plpsc.pok.ibm.com`. The following example shows defining a resource.

   Example:

   ```
   RDEFINE SERVAUTH IRR.HOST.PLPSC.POK.IBM.COM UACC(NONE)
   ```

   ____

4. Permit your web server to access this resource with READ authority. Be sure that the web server is defined as a RACF user.

   Example:

   ```
   PERMIT IRR.HOST.PLPSC.POK.IBM.COM CLASS(SERVAUTH) ID(WEBSRV) ACCESS(READ)
   ```

   ____

5. Activate the SERVAUTH class, if not already active.

   Example:

   ```
   SETROPTS CLASSACT(SERVAUTH)
   ```

   If already active, refresh the SERVAUTH class.

   Example:

   ```
   SETROPTS CLASSACT(SERVAUTH) REFRESH
   ```

   ____

Note: On a z/OS system, a `HostIdMappings` extension is not honored if the target user ID was created after the start of the validity period for the certificate containing the `HostIdMappings` extension. Therefore, if you are creating user IDs specifically for certificates with `HostIdMappings` extensions, make sure that you create the user IDs before the certificate requests are submitted. Alternately, when approving the certificate, you can modify the date that the certificate becomes valid so that it is not earlier than the date the user ID was created. For renewed certificates, all of the original information is replicated in the new certificate, including the date that the certificate becomes valid and any

HostIdMappings. If you want to change a HostIdMappings extension when approving the renewed certificate, you must also modify the date that the certificate becomes valid so that it is not earlier than the date the user ID was created.

See *z/OS Security Server RACF Command Language Reference* for details about syntax and authorization that is required for using the RACDCERT command.

Locating your PKI Services certificates and key ring

The IKYSETUP exec sets up the RACF environment for PKI Services. After the setup is complete, you might need to go back and locate the PKI Services CA certificate, key ring, or the optional RA certificate, possibly to diagnose error conditions. You can do this by using various RACF TSO commands.

Before you begin

You need to determine the following setup information:

Table 93. Information you need for locating your PKI Services certificates and key ring

Information needed	Where to find this information	Record your value here
ca_label - The label of your CA certificate in RACF	See Table 11 on page 41.	
ra_label - The label of your RA certificate in RACF	See Table 11 on page 41.	
ca_ring - The PKI Services SAF key ring	See Table 19 on page 55.	
daemon - The user ID for the PKI Services daemon	See Table 19 on page 55.	
log_dsn - The data set name of the IKYSETUP log	See Table 19 on page 55.	
cacert_dsn - The data set name of your CA certificate as exported from RACF	See Table 19 on page 55.	

Steps for locating the PKI Services certificates and key ring

Perform the following steps to locate the PKI Services CA certificate, key ring, and the optional RA certificate:

1. Locate the CA certificate using one of the following two methods (Step “1.a” on page 484 or Step “1.b” on page 485) and examine its information.

   a. Locate the CA certificate using the name of its export data set. (Get the export data set name from *cacert_dsn* in Table 93 on page 484.) Display its information by executing the following RACF command from a TSO command prompt:

   ```
   RACDCERT CHECKCERT(cacert_dsn)
   ```

   Sample output:

   ```
   Digital certificate information for CERTAUTH:
     Label: Local PKI CA
     Certificate ID: 2QiJmZmDhZmjgdOWg4GTQNfSyUDDwUBA
     Status: HIGHTRUST
     Start Date: 2001/06/04 23:00:00
     End Date:   2020/01/01 22:59:59
     Serial Number:
          >00<
     Issuer's Name:
   ```

```
     >OU=Human Resources Certificate Authority.O=IBM.C=US<
Subject's Name:
     >OU=Human Resources Certificate Authority.O=IBM.C=US<
Key Usage: CERTSIGN
Key Type: RSA
Key Size: 1024
Private Key: Yes
```

b. Alternately, locate the CA certificate using its certificate label. (Get the label name from *ca_label* in Table 93 on page 484.) Display its information by entering the following RACF command from a TSO command prompt.

```
RACDCERT CERTAUTH LIST(LABEL('ca_label'))
```

The RACDCERT CERTAUTH LIST command produces the same output as the RACDCERT CHECKCERT (shown in Step "1.a" on page 484) with the addition of information about any ring associations. For example:

Sample output:

```
Ring Associations:
  Ring Owner: PKISRVD
  Ring:
     >CAring<
```

c. Examine the CA certificate information. If you are diagnosing errors, note the following:

- The first line must indicate that this is a CERTAUTH certificate.
- `Label` must match your `ca_label` value (as in the preceding table).
- If `Serial Number` is not equal to `00`, this indicates that the certificate has been renewed or was issued by another certificate authority.
- If `Issuer's Name` differs from `Subject's Name`, this indicates that the certificate was issued by another certificate authority.
- `Subject's Name` must match the original value recorded for the PKI Services SUBJECTSDN in the IKYSETUP log.
- `Private Key` must show YES.
- `Key Type` indicates whether the key is an RSA, DSA, NISTECC, or BPECC key.
- If `Ring Associations` are listed, ensure that an association is displayed for the daemon user ID as ring owner and your `ca_ring` value (from Table 93 on page 484) as ring name.

---

2. Locate the CA key ring and examine its information.

a. Get the ring name from *ca_ring* in Table 93 on page 484 and display its information by executing the following RACF command from a TSO command prompt:

```
RACDCERT ID(daemon) LISTRING(ca_ring)
```

Sample output:

```
Digital ring information for user PKISRVD:

  Ring:
       >CAring<
  Certificate Label Name             Cert Owner     USAGE      DEFAULT
  --------------------------------   ------------   --------   -------
  Local PKI CA                       CERTAUTH       PERSONAL     YES
  Local PKI RA                       PKISRVD        PERSONAL     NO
```

b. Examine the key ring information. If you are diagnosing errors, note the following:

- The entry for the PKI Services CA certificate must have USAGE PERSONAL and DEFAULT YES.
- If you use an optional RA certificate, you see the second line. If present, the entry for the PKI Services RA certificate must have USAGE PERSONAL and DEFAULT NO.

---

3. If you use an optional RA certificate, locate it and examine its information.

   a. Locate the RA certificate using its certificate label. (Get the RA's certificate label from *ra_label* in Table 93 on page 484 or from the RACDCERT LISTRING output shown in Step "2.a" on page 485.) Display the RA certificate information by executing the following RACF command from a TSO command prompt:

   ```
   RACDCERT ID(certificate-owner) LIST(LABEL('certificate-label-name'))
   ```

   Sample output:

   ```
   Digital certificate information for PKISRVD:
     Label: Local PKI RA
     Certificate ID: 2QiJmZmDhZmjgdOWg4GTQNfSyUDDwUBA
     Status: TRUST
     Start Date: 2001/06/04 23:00:00
     End Date:   2020/01/01 22:59:59
     Serial Number:
          >01<
     Issuer's Name:
          >OU=Human Resources Certificate Authority.O=IBM.C=US<
     Subject's Name:
          >CN=Registration Authority.OU=Human Resources Certif<
          >icate Authority.O=IBM.C=US<
     Key Usage: HANDSHAKE
     Key Type: RSA
     Key Size: 1024
     Private Key: Yes
   ```

   b. Examine the RA certificate information. If you are diagnosing errors, note the following:

   - The user ID of the certificate owner (indicated in the first line) must match the user ID of the PKI Services daemon.
   - `Issuer's Name` must match the `Subject's Name` of the CA certificate.
   - `Private Key` must show YES.
   - `Key Type` indicates the key is an RSA key.

Establishing PKI Services as an intermediate CA

The default setup for PKI Services establishes the PKI Services certificate authority as a root CA, also known as a self-signed CA. Because there is no established trust hierarchy leading to a self-signed certificate, it is impossible to verify that a self-signed certificate is genuine. Accordingly, any person or application that wants to process certificates issued by a root authority must explicitly trust the authenticity of the self-signed CA certificate.

Alternately, you can establish the PKI Services certificate authority as an intermediate (subordinate) certificate authority. An intermediate certificate authority is one whose certificate is signed by another higher certificate authority. This higher certificate authority can be a root CA or another intermediate CA. If the root CA certificate has previously been trusted, you can verify any lower intermediate CA certificate using the higher certificate.

In the following steps, you are replacing the self-signed CA certificate created by IKYSETUP with one signed by another authority.

Steps for changing PKI Services from a self-signed CA to an intermediate CA

Before you begin

1. This procedure assumes that the PKI Services CA certificate is issued by a root, or self-signed, CA.

2. The commands in the steps that follow include several variables. The following table describes these variables. Determine the values for these variables and record the information in the blank boxes:

Table 94. Information you need for establishing PKI Services as an intermediate CA

Information needed	**Where to find this information**	**Record your value here**
cacert_dsn - The data set name of the new PKI Services CA certificate.		
ca_label - The label of your CA certificate in RACF	See Table 11 on page 41.	
export_dsn - The data set name of the root CA certificate as exported from RACF.		
temp_dsn - The name of the temporary data set to contain your new certificate request and returned certificate.	You decide this based on local data set naming conventions.	

Procedure

Perform the following steps to change PKI Services from a self-signed certificate authority to an intermediate certificate authority:

1. Determine what certificate authority is acting as a higher authority for PKI Services. (This could be a public certificate authority, such as VeriSign, or a local, internal certificate authority, even another instance of PKI Services.)

   ____

2. Create a new certificate request from your existing self-signed CA certificate by entering the following RACF command from a TSO command prompt:

   ```
   RACDCERT CERTAUTH GENREQ(LABEL('ca_label')) DSN(temp_dsn)
   ```

   ____

3. Send the certificate request to the higher certificate authority, following the procedures that the higher authority requires.

   ____

4. If the root CA is not one that is already known by RACF, then add the root CA to RACF as a certificate authority. To do this:

   a. Receive the root CA certificate and place it into the certificate data set (*temp_dsn*).

      Note: The procedure for doing this can vary greatly depending on how the higher certificate authority delivered the certificate:

      - If the certificate is delivered as base64 encoded text, the easiest way to deposit the certificate into the data set is to edit the certificate data set:

        1) Delete all existing lines in *temp_dsn*.
        2) Copy the base64 encoded text.
        3) Paste the copied text into the ISPF edit window.
        4) Save.

      - If the certificate is delivered as binary data (also called DER encoded), the easiest way to deposit the certificate into the data set is to use binary FTP.

b. Add the new root CA certificate into the RACF database by entering the following RACF command from a TSO command prompt:

```
RACDCERT CERTAUTH ADD(temp_dsn) WITHLABEL('label-for-root-CA')
```

____

5. Add the new PKI Services CA to RACF as a certificate authority:

   a. Receive the PKI Services CA certificate and place it into the certificate data set (*cacert_dsn*). This step is similar to step "4" on page 487, except that it uses *cacert_dsn* as the data set name instead of *temp_dsn*, because you want to keep the PKI Services CA certificate permanently in the data set *cacert_dsn*.

   b. Add the new PKI Services CA certificate back into the RACF database by entering the following RACF command from a TSO command prompt:

   ```
   RACDCERT CERTAUTH ADD(cacert_dsn)
   ```

   Guideline: Do not specify a label on this command.

____

6. Export the root CA certificate in DER format to the export data set by entering the following RACF command from a TSO command prompt:

```
RACDCERT CERTAUTH EXPORT(LABEL('label-for-root-CA')) DSN(export_dsn)
   FORMAT(CERTDER)
```

____

7. Make your new root CA certificate available to your clients, because it becomes the web server's root CA certificate too. To do this, set up the `var` directory by performing Step "2" on page 95 through Step "4" on page 96 in " Steps for setting up the var directory" on page 95.

   Note: Make sure that the root CA certificate, not your intermediate CA certificate, is stored in `/var/pkiserv/cacert.der`.

____

Renewing your PKI Services CA and RA certificates

Eventually, your PKI Services CA and RA certificates expires. To avoid complications that are related to an expired CA or RA certificate, renew those certificates before they expire. (You receive an MVS console message IKYP026E as the expiration date approaches.)

This topic contains these procedures:

- "Steps for renewing your PKI Services CA certificate" on page 488
- "Steps for renewing your PKI Services RA certificate" on page 490

Steps for renewing your PKI Services CA certificate

Before you begin

The commands in the steps that follow include several variables. The following table describes these variables. Determine the values for these variables and record the information in the blank boxes:

Table 95. Information you need for renewing your PKI Services certificate authority certificate

Information needed	Where to find this information	Record your value here
cacert_dsn - The data set name of your renewed CA certificate as exported from RACF. (This data set is needed for recovery.)		
ca_label - The label of your CA certificate in RACF	See Table 11 on page 41.	
temp_dsn - The temporary data set to contain your new certificate request and returned certificate.	You decide this based on local data set naming conventions.	

Procedure

Perform the following steps to renew your PKI Services CA certificate:

1. Create a new certificate request from your current CA certificate by entering the following RACF command from a TSO command prompt:

```
RACDCERT CERTAUTH GENREQ(LABEL('ca_label')) DSN(temp_dsn)
```

2. If your PKI Services certificate authority is a root CA (that is, it has a self-signed certificate, which is the default), generate the self-signed renewal certificate by entering the following RACF command from a TSO command prompt. The *ca_expires* variable indicates the new expiration date.

```
RACDCERT CERTAUTH GENCERT(temp_dsn) NOTAFTER(DATE(ca_expires))
  SIGNWITH(CERTAUTH LABEL('ca_label'))
```

3. Alternately, if your PKI Services certificate authority is an intermediate certificate authority, perform the following steps:

   a. Send the certificate request to the higher (external) CA, following the procedures that the higher authority requires. If your CA retains the original certificate signing requests (CSR), you might not need to create and store a new request based on the expiring certificate. You might be able to request a renewal using the original CSR.

   b. After the certificate has been issued, receive the certificate back into the certificate data set (*temp_dsn*).

      Note: The procedure for doing this can vary greatly depending on how the higher certificate authority delivers the new certificate:

      - If the certificate is delivered as base64 encoded text, the easiest way to deposit the certificate into the data set is to edit the certificate data set:
        1) Delete all existing lines in *temp_dsn*.
        2) Copy the base64 encoded text.
        3) Paste the copied text into the ISPF edit window.
        4) Save.
      - If the certificate is delivered as binary data (also called DER encoded), the easiest way to deposit the certificate into the data set is to use binary FTP.

   c. Add the renewed certificate back into the RACF database by entering the following RACF command from a TSO command prompt:

```
RACDCERT CERTAUTH ADD(temp_dsn)
```

      Do not specify a label on this command.

4. Export the certificate in DER format to the CA certificate data set by entering the following RACF command from a TSO command prompt:

   ```
   RACDCERT CERTAUTH EXPORT(LABEL('ca_label')) DSN(cacert_dsn) FORMAT(CERTDER)
   ```

   Save this data set for recovery if needed later.
5. If your PKI Services certificate authority is a root CA, *and* it is also the web server's root certificate, the renewed root needs to be accessible to the clients. To make your new certificate available to your clients, set up the `/var/pkiserv` directory by performing Step "2" on page 95 through Step "4" on page 96 in " Steps for setting up the var directory" on page 95.
6. Stop and restart PKI Services.

Steps for renewing your PKI Services RA certificate

Before you begin

The commands in the steps that follow include several variables. The following table describes these variables. Determine the values for these variables and record the information in the blank boxes:

Table 96. Information you need for renewing your PKI Services RA certificate

Information needed	Where to find this information	Record your value here
ca_label - The label of your CA certificate in RACF	See Table 11 on page 41.	
daemon - The user ID of the PKI Services daemon.	See Table 19 on page 55.	
racert_dsn - The name of the data set to contain your new certificate request.	You decide this based on local data set naming conventions.	
ra_label - The label of your RA certificate in RACF.	See Table 11 on page 41.	

Procedure

Perform the following steps to renew your PKI Services RA certificate:

1. Create a new certificate request from your existing RA certificate by entering the following RACF command from a TSO command prompt:

   ```
   RACDCERT ID(daemon) GENREQ(LABEL('ra_label')) DSN(racert_dsn)
   ```

2. Create the renewed PKI Services certified RA certificate by entering the following RACF command from a TSO command prompt. The *ra_expires* variable indicates the new expiration date.

   ```
   RACDCERT ID(daemon) GENCERT(racert_dsn) NOTAFTER(DATE(ra_expires))
     SIGNWITH(CERTAUTH LABEL('ca_label'))
   ```

3. Stop and restart PKI Services.

Recovering a CA certificate profile

Unless you change the IKYSETUP REXX exec to disable the function, IKYSETUP automatically backs up the PKI Services CA certificate and private key to a data set that has PKCS #12 format, if the certificate and private key were created by software. (If the certificate was created by hardware, for example by RACDCERT with the PKDS or TKDS keyword, it is not backed up.) If the CA certificate profile in the RACF database is accidentally deleted, you can recover it by adding the certificate and private key back to the RACF database from the backup PKCS #12 data set.

Steps for recovering a CA certificate profile

Before you begin

The commands in the steps that follow include several variables. Table 97 on page 491 describes these variables. Determine the values for these variables and record the information in the blank boxes:

Table 97. Information you need for recovering a CA certificate profile

Information needed	Where to find this information	Record your value here
backup_dsn - The name of the data set containing the backup copy of your original CA certificate and its private key	See Table 19 on page 55.	
cacert_dsn - The data set name of your CA certificate as exported from RACF		
ca_label - The label of your CA certificate in RACF	See Table 11 on page 41.	
ca_ring - The PKI Services SAF key ring	See Table 19 on page 55.	
daemon - The user ID for the PKI Services daemon	See Table 19 on page 55.	
your-passphrase - The passphrase you used when backing up the private key	You specified this when running IKYSETUP.	

Procedure

Perform the following steps to recover a CA certificate profile:

1. Issue the following TSO commands:

   Notes:

   a. If your CA certificate has been renewed, the second ADD command recovers the most current version using the saved CA certificate. If your certificate has not been renewed, you can omit the second ADD command. For information about renewing your CA certificate, see "Renewing your PKI Services CA and RA certificates" on page 488.

```
RACDCERT CERTAUTH ADD(backup_dsn) PASSWORD(your-passphrase)
     WITHLABEL('ca_label') ICSF
RACDCERT CERTAUTH ADD(cacert_dsn)
RACDCERT ID(daemon) CONNECT(CERTAUTH LABEL('ca_label')
     RING(ca_ring) USAGE(PERSONAL) DEFAULT)
```

Retiring and replacing the PKI Services CA private key

For certificates that are associated with private keys, such as the PKI Services CA certificate, you should periodically retire the private keys and replace them with new ones. Do this to prevent private keys from being overused. (The more a key is used, the more susceptible it is to being broken and recovered by an unintended party.) This process is commonly called *certificate rekeying* or *key rollover*.

You may also need to perform *certificate rekeying* when you want to run PKI Services in FIPS mode, or when you want to reconfigure PKI Services to run in a higher FIPS mode. See "Steps to retire and replace the PKI Services non-FIPS compliant CA private key for the PKI templates" on page 493.

To rekey and rollover the PKI Services private key, use the REKEY and ROLLOVER operands of the RACF RACDCERT command. The REKEY operand makes a self-signed copy of the original certificate with a new public-private key pair. The ROLLOVER operand finalizes the rekey operation by replacing the use of the original certificate with the new certificate in every key ring to which the original certificate is connected. It also destroys the original private key and copies over information about its serial number base so the new certificate can be used to sign new certificates.

A retired CA certificate cannot be used to sign new certificates. However, until it expires, it can be used to verify previously signed certificates. If you have an RA certificate for SCEP processing that was not replaced when the CA certificate that signed it was retired, you need to reconnect the CA certificate to the CA key ring.

Steps to retire and replace the PKI Services CA private key for the PKI templates

The commands that are used in this procedure are examples that are based on the following scenario:

Assumptions:

- The certificate that you are rekeying is a CERTAUTH certificate with label `'Local PKI CA'`. It was issued by a commercial CA and is being used by PKI Services for the PKI templates as a certificate authority (CA) certificate, making the PKI Services CA a subordinate CA.
- The ICSF CCA cryptographic coprocessor be used to generate the new key-pair and store in the Public Key Data Set (PKDS)
- The new private key is 2048-bits RSA key (RACF default size).

Perform the following procedure to rekey and replace the private key.

1. Initiate the rekeying by executing the following RACF command:

   ```
   RACDCERT CERTAUTH REKEY(LABEL('Local PKI CA')) WITHLABEL('Local PKI
   CA-2')
   RSA(PKDS(*))
   ```

   ____

2. Create a request for a commercial CA to sign the new public key and reissue the certificate. To create a certificate request for the new key and store it in MVS data set `'SYSADM.CERT.REQ'`, issue the following command:

   ```
   RACDCERT CERTAUTH GENREQ(LABEL('Local PKI CA-2')) DSN('SYSADM.CERT.REQ')
   ```

   Restriction: The certificate request data that is contained in the data set must be sent to, and received from, the commercial CA using the process defined by the CA. Those steps are not included.

   ____

3. Receive the newly signed and reissued certificate back from the commercial CA into MVS data set `'SYSADM.CERT.B64'`.

   ____

4. Add the newly signed certificate into RACF and replace the self-signed rekeyed one by executing the following command:

   ```
   RACDCERT CERTAUTH ADD('SYSADM.CERT.B64')
   ```

   ____

5. You are now ready to retire the original certificate and must stop all use of the original private key. Stop the PKI Services daemon.

   Note: At this point, the original certificate and its private key label exist in RACF with label `'Local PKI CA'`. The new certificate and its private key label exist in a separate entry in RACF with label `'Local PKI CA-2'`. You can proceed to rollover the key.

   ____

6. Finalize the rollover by entering the following command:

   ```
   RACDCERT CERTAUTH ROLLOVER(LABEL('Local PKI CA')) NEWLABEL('Local PKI CA-2')
   ```

   ____

7. If an RA certificate is in use that was signed by the retired CA certificate, connect the retired CA certificate to the key ring.

   ```
   RACDCERT ID(daemon) CONNECT(CERTAUTH LABEL('Local PKI CA') RING(ringname) USAGE(CERTAUTH))
   ```

8. Restart the PKI Services daemon.

   ____

When you are done: You have retired and replaced the old PKI Services CA certificate. All the information for the original certificate is updated to reflect the new certificate, including the key ring connections. You can now use the new certificate and its private key. You can continue to use the old certificate for signature verification purposes until it expires. However, you cannot use the old certificate to sign new certificates. Additionally, do not connect the old certificate to any key rings as the default certificate.

Steps to retire and replace the PKI Services non-FIPS compliant CA private key for the PKI templates

The commands that are used in this procedure are examples that are based on the following scenario:

Assumptions:

- The certificate that you are rekeying is a CERTAUTH certificate with label `'Local PKI CA'`. It was a self-signed certificate with 2048-bit RSA key stored in PKDS.
- You want the new key size to be 4096.
- You want the new certificate to expire in 2037-12-31.
- You want the new key pair be generated using the ICSF PKCS#11 cryptographic coprocessor and you already set up the token with the name `PKISRVD.PKITOKEN`.

Perform the following procedure to rekey and replace the private key.

1. Initiate the rekeying by executing the following RACF command:

   ```
   RACDCERT CERTAUTH REKEY(LABEL('Local PKI CA')) WITHLABEL('Local PKI CA-2')
   RSA(TOKEN(PKISRVD.PKITOKEN)) SIZE(4096) NOTAFTER (DATE(2037-12-31))
   ```

   Note: At this point, the original certificate and its private key label exist in RACF with label `'Local PKI CA'`. The new certificate and its private key label exist in a separate entry in RACF with label `'Local PKI CA-2'`. You can proceed to rollover the key.

   ____

2. Stop the PKI Services daemon.

   ____

3. Perform rollover by entering the following command:

   ```
   RACDCERT CERTAUTH ROLLOVER(LABEL('Local PKI CA')) NEWLABEL('Local PKI CA-2')
   ```

   ____

4. Restart the PKI Services daemon.

Steps to retire and replace the PKI Services CA private key for the SAF templates: Scenario 1

The commands that are used in this procedure are examples that are based on the following scenario:

Assumptions:

- The certificate that you are rekeying is a CERTAUTH certificate with label `'taca'`.
- It was issued by a local CA certificate that is labeled `'Local RACF CA'` that was generated by RACF and is being used by PKI Services for the SAF templates as a certificate authority (CA) certificate.

Perform the following procedure to rekey and replace the private key.

1. Initiate the rekeying by executing the following RACF command:

   ```
   RACDCERT CERTAUTH REKEY(LABEL('taca')) WITHLABEL('taca-2')
   ```

   ____

2. Generate a certificate request that is based on the new self-signed certificate and store it in MVS data set `'SYSADM.CERT.REQ'` by executing the following command:

   ```
   RACDCERT CERTAUTH GENREQ(LABEL('taca-2')) DSN('SYSADM.CERT.REQ')
   ```

   ____

3. Issue the following command to sign the new certificate, specifying the planned expiration date in the format expected by RACDCERT, for example, YYYY-MM-DD::

   ```
   RACDCERT CERTAUTH GENCERT('SYSADM.CERT.REQ') NOTAFTER(DATE(<date the cert will expire>))
   SIGNWITH(CERTAUTH LABEL('Local RACF CA'))
   ```

   At this point, the original certificate and its private key exist in RACF with the label `'taca'`. The new certificate and its private key exist in a separate entry in RACF with the label `'taca-2'`. You can proceed to rollover the key.

   ____

4. Finalize the rollover by entering the following command:

   ```
   RACDCERT CERTAUTH ROLLOVER(LABEL('taca')) NEWLABEL('taca-2')
   ```

   ____

5. Change the certificate label that is used in the SIGNWITH field in the SAF templates to the new label name.

   ____

When you are done: You have retired and replaced the old certificate. All the information for the original certificate is updated to reflect the new certificate, including the key ring connections. You can now use the new certificate and its private key. You can continue to use the old certificate for signature verification purposes until it expires. However, you cannot use the old certificate to sign new certificates. Additionally, do not connect the old certificate to any key rings as the default certificate.

Steps to retire and replace the PKI Services CA private key for the SAF templates: Scenario 2

The commands that are used in this procedure are examples that are based on the following scenario:

Assumptions:

- The certificate that you are rekeying is a CERTAUTH certificate with label `'taca'`.
- It was a self-signed certificate in RACF and is being used by PKI Services for the SAF templates as a certificate authority (CA) certificate.

Perform the following procedure to rekey and replace the private key.

1. Initiate the rekeying by executing the following RACF command:

   ```
   RACDCERT CERTAUTH REKEY(LABEL('taca'))
      WITHLABEL('taca-2')
   ```

At this point, the original certificate and its private key exist in RACF with the label `'taca'`. The new certificate and its private key exist in a separate entry in RACF with the label `'taca-2'`. You can proceed to rollover the key.

____

2. Finalize the rollover by entering the following command:

```
RACDCERT CERTAUTH ROLLOVER(LABEL('taca')) NEWLABEL('taca-2')
```

____

3. Change the certificate label that is used in the SIGNWITH field in the SAF templates to the new label name.

____

When you are done: You have retired and replaced the old certificate. All the information for the original certificate is updated to reflect the new certificate, including the key ring connections. You can now use the new certificate and its private key. You can continue to use the old certificate for signature verification purposes until it expires. However, you cannot use the old certificate to sign new certificates. Additionally, do not connect the old certificate to any key rings as the default certificate.

R_PKIServ (IRRSPX00 and IRRSPX64) callable service

Authorized applications, such as servers, that invoke the `R_PKIServ` callable service (IRRSPX00 for 31 bit and IRRSPX64 for 64 bit) can request the generation, retrieval, and administration of PKIX-compliant X.509 Version 3 certificates and certificate requests. Applications can request end-user functions or administrative functions related to these requests. You authorize these applications by administering RACF resources in the FACILITY class, which is based on whether the application requests end-user functions or administrative functions.

You can authorize access to administrative functions on a more granular level using resources in the RACF PKISERV class. The PKISERV class profiles are checked in addition to the FACILITY class profiles.

See *z/OS Security Server RACF Callable Services* for the details of invoking IRRSPX00.

Authorizing end-user functions

The end-user functions are:

EXPORT
Retrieves (exports) a previously requested certificate, or retrieves (exports) the PKI Services registration authority (RA) certificate or the certificate authority (CA) certificate.

GENCERT
Generates an auto-approved certificate.

GENRENEW
Generates an auto-approved renewal certificate. (The request submitted is automatically approved.)

QRECOVER
Lists certificates whose key pairs were generated by PKI Services under a requestor’s email address and passphrase.

REQCERT
Requests a certificate that an administrator must approve before it is created.

REQRENEW
Requests certificate renewal. The administrator needs to approve the request before the certificate is renewed.

RESPOND
Invokes the PKI OCSP responder.

REVOKE
Revokes a certificate that was previously issued.

SCEPREQ
Generates a certificate request using Simple Certificate Enrollment Protocol (SCEP).

VERIFY
Confirms that a given user certificate was issued by this certificate authority and, if so, returns the certificate fields.

For end-user functions, FACILITY class resources protect this interface. Access authority is based on the user ID for the application (the user ID from the ACEE associated with the address space). To determine the user ID for the application, the current TCB is checked for an ACEE. If one is found, the authority of that user is checked. If there is no ACEE associated with the current TCB, the ACEE associated with the address space is used to locate the user ID.

The form for the FACILITY class resources is:

```
IRR.RPKISERV.function[.ca_domain]
```

function
Specifies one of the end-user function names in the preceding list.

ca_domain
Optionally specifies the PKI Services certificate authority (CA) domain name. Use this when your installation has established multiple PKI Services CAs and the `CA_domain` parameter is provided with IRRSPX00 for 31 bit and IRRSPX64 for 64 bit.

Restriction: If the name of your initial CA domain is longer than 8 characters, you must truncate it to exactly 8 characters when you define the resource name in the FACILITY class.

Example: For the GENCERT function, when the *ca_domain* is named `Customers` and the `CA_domain` parameter is provided with IRRSPX00 for 31 bit and IRRSPX64 for 64 bit, then the FACILITY class resource controlling the function is IRR.RPKISERV.GENCERT.CUSTOMER. (The name `Customers` was truncated to CUSTOMER. See the restriction for the *ca_domain* parameter.) When the `CA_domain` parameter is not provided with IRRSPX00 for 31 bit and IRRSPX64 for 64 bit, the FACILITY class resource is IRR.RPKISERV.GENCERT.

The access authorities you can assign for these FACILITY class resources have the following effects:

NONE
Access is denied.

READ
Access is permitted based on subsequent access checks against the caller's user ID.

UPDATE
Access is permitted based on subsequent access checks against the application's user ID.

CONTROL (or user ID has RACF SPECIAL)
Access is permitted, and no subsequent access checks are made.

Example: If you defined the FACILITY class profile IRR.RPKISERV.GENCERT.CUSTOMER to control access to the GENCERT function on the CA domain named `Customers`, you can prevent the user ID MYAPP from using the GENCERT function on that CA domain by issuing the command:

```
PERMIT IRR.RPKISERV.GENCERT.CUSTOMER CLASS(FACILITY) ID(MYAPP) ACCESS(NONE)
```

For SAF GENCERT and EXPORT requests where the application has READ and UPDATE access, subsequent access checks are performed against the `IRR.DIGTCERT.function` FACILITY resources. These are identical to the checks the RACDCERT TSO command makes. See *z/OS Security Server RACF Command Language Reference* for more information.

For PKI Services EXPORT, GENCERT, GENRENEW, QRECOVER, REQCERT, REQRENEW, RESPOND, REVOKE, SCEPREQ, and VERIFY requests in which the application has READ and UPDATE access, subsequent access checks are performed against the `IRR.DIGTCERT.function` FACILITY resources.

The following table summarizes the access requirements for the user ID whose access is checked.

Table 98. Summary of access authorities required for PKI Services requests

Request	Access
EXPORT	• IRR.DIGTCERT.EXPORT – READ access if `PassPhrase` is specified or if `CertID` is specified as `PKICACERT`. – UPDATE access if the `PassPhrase` parameter is not specified with IRRSPX00 for 31 bit and IRRSPX64 for 64 bit. – CONTROL access if you want to export a PKCS #7 certificate.
GENCERT	• IRR.DIGTCERT.GENCERT — CONTROL access • IRR.DIGTCERT.ADD – UPDATE access if any hostIdMappings information is specified in the certificate request parameter list or the `UserId` field in the certificate request parameter list indicates the certificate is being requested for another user other than the caller – READ access otherwise
GENRENEW	• IRR.DIGTCERT.GENRENEW — READ access • IRR.DIGTCERT.GENCERT — CONTROL access **Note:** It is assumed that the calling application has already verified the input certificate using the VERIFY function.
QRECOVER	• IRR.DIGTCERT.QRECOVER — READ access
REQCERT	• IRR.DIGTCERT.REQCERT — READ access
REQRENEW	• IRR.DIGTCERT.REQRENEW — READ access **Note:** It is assumed that the calling application has already verified the input certificate using the VERIFY function.
RESPOND	• IRR.DIGTCERT.RESPOND — READ access
REVOKE	• IRR.DIGTCERT.REVOKE — READ access **Note:** It is assumed that the calling application has already verified the target certificate using the VERIFY function.
SCEPREQ	• IRR.DIGTCERT.SCEPREQ — READ access
VERIFY	• IRR.DIGTCERT.VERIFY — READ access **Note:** It is assumed that the calling application has already verified that the end user possesses the private key that correlates to the input certificate.

Authorizing administrative functions

The administrative functions are:

CERTDETAILS
Get detailed information about one PKI Services issued certificate.

MODIFYCERTS
Change PKI Services issued certificates.

MODIFYREQS
Change PKI Services certificate requests.

QUERYCERTS
Query PKI Services issued certificates.

QUERYREQS
Query PKI Services about certificate requests.

PREREGISTER
Preregister clients who use Simple Certificate Enrollment Protocol (SCEP) or Enrollment over Secure Transport Protocol (EST).

REQDETAILS
Get detailed information about one PKI Services certificate request.

To control access to these functions:

- Use resources in the RACF FACILITY class. This class allows you to control access based on the CA domain.
- Use resources in the RACF PKISERV class. This class allows you to control access on a more granular level than the FACILITY class, which is based on the CA domain, the administrative function, and the template.

Using the FACILITY class to control access to administrative functions

For the all administrative functions, the following single FACILITY class resource protects this interface.

```
IRR.RPKISERV.PKIADMIN[.ca_domain]
```

ca_domain
Optionally specifies the PKI Services certificate authority (CA) domain name. Use this when your installation has established multiple PKI Services CAs and the `CA_domain` parameter is provided with IRRSPX00 for 31 bit and IRRSPX64 for 64 bit.

Restriction: If the name of your initial CA domain is longer than 8 characters, you must truncate it to exactly 8 characters when you define the resource name in the FACILITY class.

- If the caller is RACF SPECIAL, no further access is necessary.
- Otherwise, the caller needs:
  - READ access to perform read operations (QUERYREQS, QUERYCERTS, REQDETAILS, and CERTDETAILS)
  - UPDATE access for the action operations (PREREGISTER, MODIFYREQS, and MODIFYCERTS).

Example: For administrative functions, when the *ca_domain* is named `Customers` and the `CA_domain` parameter is provided with IRRSPX00 for 31 bit and IRRSPX64 for 64 bit, the FACILITY class resource controlling this interface is IRR.RPKISERV.PKIADMIN.CUSTOMER. (The name `Customers` was truncated to CUSTOMER. See the restriction for the *`ca_domain`* value.) When the `CA_domain` parameter is not provided with IRRSPX00 for 31 bit and IRRSPX64 for 64 bit, IRR.RPKISERV.PKIADMIN is the name of the FACILITY class resource.

To determine the appropriate access level of the caller, the current TCB is checked for an ACEE. If one is found, the authority of that user is checked. If there is no ACEE associated with the current TCB, the ACEE associated with the address space is used to locate the user ID.

Attention: UPDATE access to the IRR.RPKISERV.PKIADMIN[.*ca_domain*] resource also controls who can act as PKI Services administrators. PKI Services administrators play a very powerful role in your organization. The decisions they make when managing certificates and certificate requests determine who accesses your computer systems and what privileges they have when doing so.

Guideline: Give UPDATE authority to only highly trusted individuals, but avoid allowing these same individuals to have direct access to the end-user functions of the `R_PKIServ` callable service described in "Authorizing end-user functions" on page 495. This helps to maintain a secure separation of duties.

Using the PKISERV class to control access to administrative functions

You can use profiles in the PKISERV class to control access to R_PKIServ administrative functions on a more granular level than you can with profiles in the FACILITY class. If the `AdminGranularControl` switch in the `pkiserv.conf` configuration file is set to T, profiles in the PKISERV class are checked in addition to profiles in the FACILITY class to determine authorization to these functions. If no profile is found protecting a function, authorization to the function fails.

To use the PKISERV class, you need to take the following steps:

1. Activate generic profile checking for the class:

   ```
   SETROPTS GENERIC(PKISERV)
   ```

2. Define profiles for the PKISERV class resources and authorize users to use the resources:

   ```
   RDEFINE PKISERV profile_name UACC(NONE)
   PERMIT profile_name CLASS(PKISERV) ID(user_ID or group) ACCESS(access_level)
   ```

3. Activate and RACLIST the class:

   ```
   SETROPTS CLASSACT(PKISERV) RACLIST(PKISERV)
   ```

Any time that you update the profiles in the class, refresh the in-storage profiles:

```
SETROPTS RACLIST(PKISERV) REFRESH
```

For the query functions (QUERYREQS, QUERYCERTS, REQDETAILS, and CERTDETAILS), the resources in the PKISERV class are of the form:

```
ca_domain.action.template_nickname
```

where

ca_domain
Specifies the PKI Services certificate authority (CA) domain name.

Rules:

- The domain name is at most 8 characters long.
- The domain name can contain only alphanumeric characters and the national characters @, #, and $.
- If there is no domain name, the qualifier must be `NOCADOMAIN`.

action
Specifies the function. It has one of the following values:

- QUERYREQS
- QUERYCERTS
- QUERYREQDETAILS
- QUERYCERTDETAILS

Rules:

- For the REQDETAILS function, if the administrator has READ access to the QUERYREQDETAILS profile, the password value is replaced by blanks before it is returned. If the administrator has UPDATE access, the password value is returned.

- For the CERTDETAILS function, if the administrator has READ access to the QUERYCERTDETAILS profile, the password value is replaced by blanks before it is returned. If the administrator has UPDATE access, the password value is returned.
- For all other functions, READ access is sufficient.

template_nickname
Specifies the nickname of the certificate template.

Rules:

- The template nickname is at most 8 characters long.
- The template nickname can contain only alphanumeric characters and the national characters @, #, and $.
- If there is no template nickname, the qualifier must be `NONICKNAME`.

Example: An administrator has either READ or UPDATE access to the FACILITY class profile `IRR.RPKISERV.PKIADMIN.MYDOMAIN` and also has READ access to the PKISERV class profiles `MYDOMAIN.QUERYREQS.1YBSSL` and `MYDOMAIN.QUERYCERTS.1YBSSL`. That administrator can perform QUERYREQS and QUERYCERTS functions on the requests and certificates created with the template "1-Year PKI SSL Browser Certificate" in the domain MYDOMAIN. If that same administrator does not have READ or UPDATE access to the PKISERV class profile `MYDOMAIN.QUERYREQS.5YSSSL`, that administrator would not be able to perform QUERYREQS functions on requests created with the template "5-Year PKI SSL Server Certificate" in the same domain.

For the update functions (MODIFYREQS, MODIFYCERTS, and PREREGISTER), the resources in the PKISERV class are of the form:

```
ca_domain.action.template_nickname
```

where

ca_domain
Specifies the PKI Services certificate authority (CA) domain name.

Rules:

- The domain name is at most 8 characters long.
- The domain name can contain only alphanumeric characters and the national characters @, #, and $.
- If there is no domain name, the qualifier must be `NOCADOMAIN`.

action
Specifies the function. It has one of the following values:

- PREGISTER
- APPROVE (for MODIFYREQS)
- APPROVEWITHMODS (for MODIFYREQS)
- REJECT (for MODIFYREQS)
- DELETEREQS (for MODIFYREQS)
- REVOKE (for MODIFYCERTS)
- DELETECERTS (for MODIFYCERTS)
- RESUME (for MODIFYCERTS)
- AUTORENEWENABLE (for MODIFYCERTS)
- AUTORENEWDISABLE (for MODIFYCERTS)
- CHANGEMAIL (for MODIFYCERTS)
- CREATECRL (for MODIFYCERTS)
- POSTCERT (for MODIFYCERTS)

template_nickname
Specifies the nickname of the certificate template.

Rules:

- The template nickname is at most 8 characters long.
- The template nickname can contain only alphanumeric characters and the national characters @, #, and $.
- The template is irrelevant for CREATECRL and POSTCERT and the template nickname is not included in the resource name for these actions.
- You must specify a template nickname for the PREREGISTER action.
- For all actions other than CREATECRL, POSTCERT, and PREREGISTER, If there is no template nickname, the qualifier must be `NONICKNAME`.

Examples:

- READ access to the profile MYDOMAIN.APPROVE.1YBSSL allows the administrator to approve requests under the template "1-Year PKI SSL Browser Certificate" in the domain MYDOMAIN.
- READ access to the profile MYDOMAIN.APPROVEWITHMODS.1YBSSL allows the administrator to modify the content of requests and then approve them under the "1-Year PKI SSL Browser Certificate" template in the domain MYDOMAIN.
- READ access to the profile MYDOMAIN.REVOKE.1YBSSL allows the administrator to revoke or suspend certificates under the "1-Year PKI SSL Browser Certificate" template in the domain MYDOMAIN.
- READ access to the profile MYDOMAIN.PREREGISTER.5YSCEPP allows the administrator to preregister requests under the "5-Year SCEP Certificate - Preregistration" template in the domain MYDOMAIN.

Using encrypted passwords for LDAP servers

PKI Services uses an LDAP directory to store certificates. LDAP requires authenticating (binding) to the directory. You can do this by using a distinguished name and passwords. Passwords for binding (to multiple LDAP directories) can be encrypted or in clear text. The UNIX programmer or LDAP programmer or both determine whether to use encrypted LDAP bind passwords. You store information about passwords in the PKI Services configuration file, `pkiserv.conf`.

If you do not need the bind password for the LDAP server to be encrypted, you specify the values for Server1, AuthName1, and AuthPwd1 in the `pkiserv.conf` configuration file. If you want the bind password for the LDAP server to be encrypted, you can use of either one of the following profiles:

- A profile named IRR.PROXY.DEFAULTS in the FACILITY class (This profile stores default binding information. It is the profile where PKI Services looks when there is no binding information.)
- A profile (you select the name) in the LDAPBIND class. (You can name this profile whatever you want if it matches the BindProfile1 value that is specified in the `pkiserv.conf` configuration file. (See Step "3" on page 112.)

Before creating either of the preceding profiles, the RACF administrator defines the LDAP.BINDPW.KEY profile in the KEYSMSTR class. This profile contains a SSIGNON segment, which holds either the masked or encrypted value for the key that encrypts passwords stored in the RACF database. Then the RACF administrator creates either of the preceding profiles with a PROXY segment that stores the binding information (the server name, bind distinguished name, and password).

Steps for using encrypted passwords

Perform the following steps to use encrypted LDAP bind passwords:

1. Define a RACF KEYSMSTR class profile by entering the following command, replacing the highlighted value with your own key:

Example:

```
RDEFINE KEYSMSTR LDAP.BINDPW.KEY SSIGNON(KEYENCRYPTED(0023528875DECFAC))
```

In this example:

- LDAP BIND passwords are masked by using a key that is saved in the KEYSMSTR class, LDAP.BINDPW.KEY.
- The key is **0023528875DECFAC**. (Replace this with your own key.)
- KEYENCRYPTED is specified (rather than KEYMASKED) because ICSF is active.

____

2. Activate the KEYSMSTR class by entering the following command:

```
SETROPTS CLASSACT(KEYSMSTR)
```

____

3. If you intend to use the LDAPBIND class, for each LDAP directory, create a RACF LDAPBIND class profile by entering the following command:

```
RDEFINE LDAPBIND MY.LDAP.SERVER1
   PROXY(LDAPHOST(ldap://some.ldap.host:389)
   BINDDN('CN=JOE USER,OU=POUGHKEEPSIE,O=IBM,C=US') BINDPW('MYPASS1')
```

Replace the highlighted parameters as follows:

a. Optionally, replace *MY.LDAP.SERVER1* with the profile name you want to use.
b. Replace *ldap://some.ldap.host:389* with your LDAP server URL. You can specify the URL with or without the preceding string "ldap:" or "ldaps:".
c. Replace *CN=JOE USER,OU=POUGHKEEPSIE,O=IBM,C=US* with the bind DN.
d. Replace *MYPASS1* with the bind password.

   Note: All bind DN qualifiers and the bind password are case-sensitive.

____

4. If you intend to use IRR.PROXY.DEFAULTS instead of the LDAPBIND class for encrypted LDAP bind passwords, issue the following command to create the profile:

```
RDEFINE FACILITY IRR.PROXY.DEFAULTS
   PROXY(LDAPHOST(ldap://some.ldap.host:389)
   BINDDN('CN=JOE USER,OU=POUGHKEEPSIE,O=IBM,C=US') BINDPW('MYPASS1')
```

Replace the highlighted parameters as follows:

a. Replace *ldap://some.ldap.host:389* with your LDAP server URL. You can specify the URL with or without the preceding string "ldap:" or "ldaps:".
b. Replace *CN=JOE USER,OU=POUGHKEEPSIE,O=IBM,C=US* with the bind DN.
c. Replace *MYPASS1* with the bind password.

   Note: All bind DN qualifiers and the bind password are case-sensitive.

____

5. Optionally, check your work by listing the segment with the RLIST command. If you are using the LDAPBIND class, issue the following command:

```
RLIST LDAPBIND MY.LDAP.SERVER1 PROXY NORACF
```

Replace `MY.LDAP.SERVER1` with the profile name you used.

Results: This command displays information like the following:

```
CLASS        NAME
LDAPBIND     MY.LDAP.SERVER1

PROXY INFORMATION
LDAPHOST= LDAP://SOME.LDAP.HOST:389
BINDDN= CN=LDAP ADMINISTRATOR,OU=POUGHKEEPSIE,O=IBM,C=US
BINDPW= YES
```

If you are using the IRR.PROXY.DEFAULTS profile of the FACILITY class, issue the following command:

```
RLIST FACILITY IRR.PROXY.DEFAULTS PROXY NORACF
```

Results: This command displays information like the following:

```
CLASS        NAME
FACILITY     IRR.PROXY.DEFAULTS

PROXY INFORMATION
LDAPHOST= LDAP://SOME.LDAP.HOST:389
BINDDN= CN=LDAP ADMINISTRATOR,OU=POUGHKEEPSIE,O=IBM,C=US
BINDPW= YES
```

____

Part 6. Using the certificate validation service

This part explains how to implement the PKI Services Trust Policy (PKITP) plug-in for OCSF.

- Chapter 23, "PKI Services Trust Policy (PKITP)," on page 507 describes the certificate validation service. It gives an overview of the OCSF plug-in PKITP, describes certificate policies and extensions, and explains additional configuration needed for PKITP and using the Trust Policy API, `CSSM_TP_PassThrough`.

Chapter 23. PKI Services Trust Policy (PKITP)

This topic:

- Provides an overview of PKITP, the PKI Services Trust Policy plug-in for OCSF
- Describes:
  - Certificate policies
  - Revoke status checking
  - Certificate extensions
  - CRL extensions and CRL entry extensions
- Explains how to perform additional OCEP configuration needed for PKITP.
- Describes `CSSM_TP_PassThrough` (the Trust Policy API).

Overview of PKITP

The PKI Services Trust Policy (PKITP) is an OCSF plug-in to perform certificate validation against a SAF key ring that contains a trusted CA or site certificate (called an *anchor certificate*) or a virtual key ring of either CERTAUTH or SITE certificates. For information about creating a SAF key ring using the RACDCERT ADDRING command, see *z/OS Security Server RACF Command Language Reference*. For information about using a virtual key ring with the `R_datalib` callable service, see *z/OS Security Server RACF Callable Services*.

PKITP supports the following two functions through the implementation of `CSSM_TP_PassThrough`:

- **CertGroupVerify**
- **FreeEvidence**

Server applications running on z/OS can use this function to verify certificates that other network entities (for example, users and other servers) present. PKI Services or other certificate authorities might have issued these certificates.

The server application must attach to and open the key ring using the OCEP DL plug-in. (For more information about OCEP and the use of SAF key rings, see *z/OS Integrated Security Services Open Cryptographic Enhanced Plug-ins Application Programming*.) The server application must also bind to any needed LDAP directories by attaching to and opening these directories using the OCSF LDAPDL plug-in. These LDAP directories can be internal corporate directories, directories of extranet business partners, directories of public certificate authorities, or combinations of these.

The following figure illustrates this diversity. The uppercase letter boxes are certificate authorities, and the lowercase letter boxes are end-entity certificates.

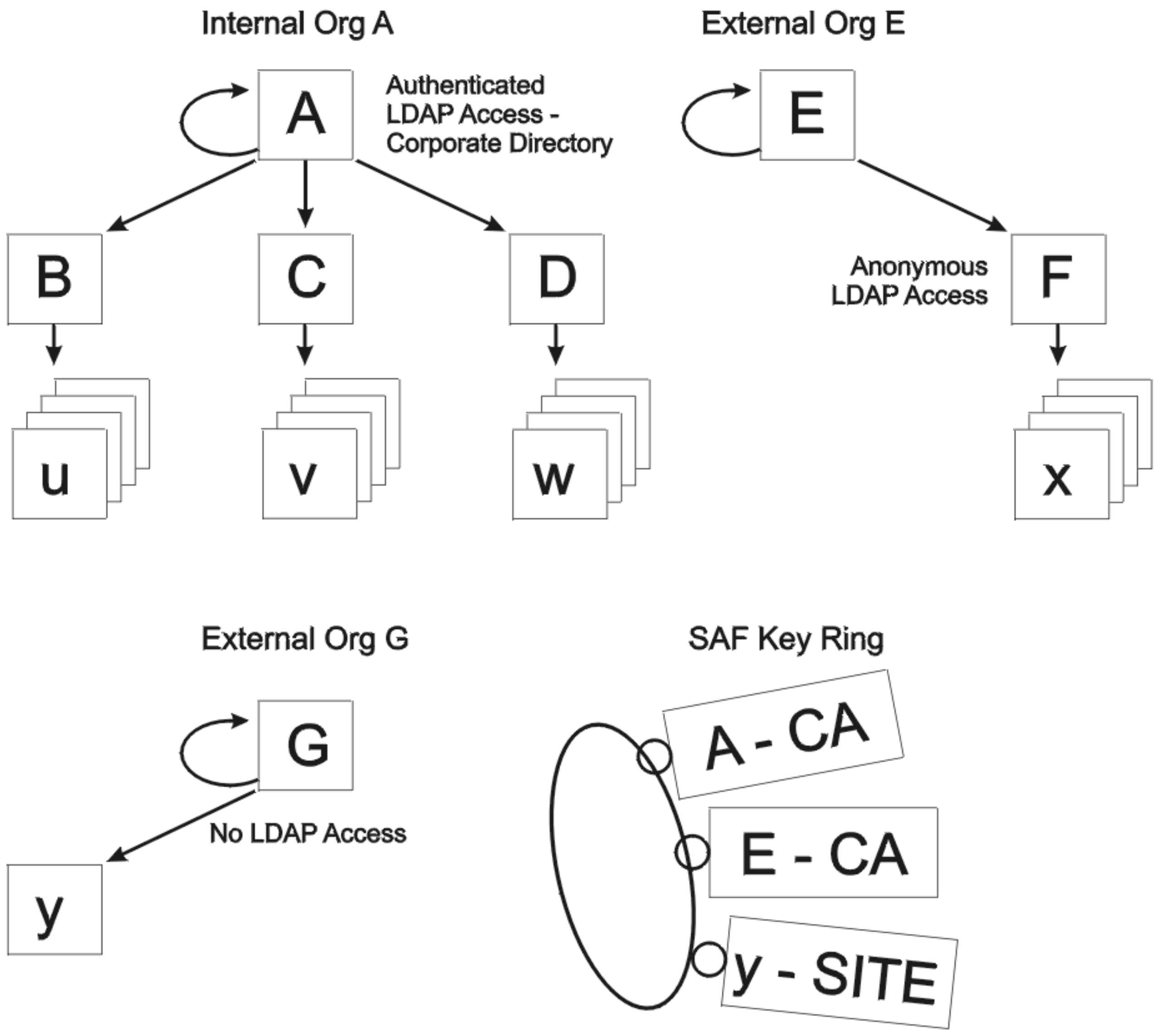


Figure 82. Examples of organizations, certificates, and chains

Organization A represents the local (corporate) certificate hierarchy. It contains one self-signed root certificate A. Perhaps RACF or the Tivoli PKI created this. B, C, and D are intermediate CAs. They could be separate instances of PKI Services. Certificates issued within this hierarchy are stored in an LDAP directory accessible to corporate server applications.

Organization E represents a public or business partner's certificate hierarchy with an LDAP directory that allows anonymous access. Organization G represents some other certificate hierarchy, in which either the directory does not exist or it is not accessible. The key ring contains three anchor certificates. Certificates A and E are trusted CAs, and there is a business need to trust end-entity certificate y, even though it cannot be verified.

If each of these CAs has posted current CRLs to either their default LDAP locations or to distribution point CRLs in LDAP and all certificate chains to be verified are genuine, the PKITP **CertGroupVerify** function can validate the following input chains:

- Single certificates `x`, `u`, `v`, or `w` (PKITP can extract the missing links from the directories.)
- Chains `u-B`, `v-C`, `w-D`, `u–B-A`, `v-C-A`, `w-D-A`, `x-F`, or `x-F-E` (These chains have no missing links.)
- Any chain beginning with certificate `y` (As Figure 82 on page 508 shows, `y` is in the key ring as a `SITE`. Site certificates are trusted regardless.)

Note that, as with the OCEP Trust Policy, non-self-signed (intermediate) CA certificates can be connected to the key ring to shorten the validation path. Doing so has the following consequences:

- Certificate revocation list (CRL) checking is not performed for the anchor certificate in the chain, even if this happens to be an intermediate CA certificate. If the intermediate CA certificate is revoked, PKITP does not detect it.
- A chain containing the parent chain of the intermediate CA cannot be verified.

  Guideline: When an intermediate CA certificate is connected to the key ring, the certificates that make up its parent chain should be connected also. This ensures that all chains originating from the intermediate CA or higher can be verified.

Certificate policies

PKITP supports CA and server application-defined certificate policies. CAs can and, in most cases, do establish their own policies for issuing certificates. These policies are declared within issued certificates through the CertificatePolicies extension. When this extension exists and is *not* marked critical, the extension is for informational purposes only - for example, specifying the URL for locating the CA's certificate practice statement (CPS). When this extension exists and is marked critical, the policies identified in the extension restrict the use of the certificate. These restrictions apply to subordinate CA certificates and to end-entity certificates. (For information about how PKI Services support the CertificatePolicies extension, see "Using certificate policies" on page 284.)

Similarly, a server application can be a general application that wants to verify certificates for no specific policy or can be an application that was written for a specific purpose and wants to verify certificates that are issued for that purpose (policy).

If the server application specifies an explicit set of policies, then at least one of these policies must be present in each certificate of the certification path (chain). Additionally, PKITP extracts the certificate polices marked critical from each certificate in the chain to determine the intersection - that is, only policies that are listed in every critically marked CertificatePolicies extension are retained. The server application must indicate that it supports at least one of these polices. If any of these tests are unsuccessful, certificate validation fails.

Checking certificate status with PKITP

PKITP checks the revocation status of a certificate by retrieving certificate revocation lists (CRLs) or, when specified in the certificate, by invoking an online validation service that uses the online certificate status protocol (OCSP).

PKITP certificate revocation checking is performed when `useCRLS` is set higher than `0`. It follows the sequence of validation stages shown in Table 99 on page 509.

Table 99. Sequence of validation stages for PKITP certificate revocation checking

Validation stage	Description
OCSP responder	The trust policy invokes the OCSP responder specified in the AuthInfoAccess extension. If none is specified or if the trust policy fails to receive certificate status from the OCSP responder, it proceeds to the next stage.
DP CRL, using the URI format	The trust policy searches for the DP CRL using the directories, if any, listed in URI format in the CRLDistributionPoints extension in the order they appear. If the DP CRL is found, it is used to determine if the certificate is revoked. If the trust policy fails to find the DP CRL using the URI formats, it proceeds to the next stage.

Table 99. Sequence of validation stages for PKITP certificate revocation checking (continued)

Validation stage	Description
DP CRL, using the distinguished-name format	The trust policy searches for the DP CRL in the LDAP directories attached through the distinguished name specified, if any, in the CRLDistributionPoints extension. If the trust policy fails to find the DP CRL using the distinguished name and the extension is *not* marked critical, it proceeds to the next stage. If the trust policy fails to find the DP CRL and the extension is marked critical, the validation fails and error code `8029` (CRL not found) is returned. If DP CRL processing is not to be performed (`useCRLS` is set to `0`) and the target certificate contains a CRLDistributionPoints extension marked critical, validation fails and error code `8029` is returned. No attempt is made to locate the DP CRL.
Global revocation list	The trust policy uses the global CRL to find revocation status information for the certificate.

Certificate extensions

PKITP supports the following certificate extensions:

AuthorityInformationAccess
Checked for form only.

AuthorityKeyIdentifier
Checked for form only.

BasicConstraints
For CA certificate, cA flag must be on. Also checked for certification path length.

CertificatePolicies
See "Certificate policies" on page 509.

CRLDistributionPoints
See "Checking certificate status with PKITP" on page 509.

HostIdMappings
Checked for form only.

IssuerAltName
Checked for form only. Must be marked critical if the issuer DN is empty.

KeyUsage
For CA certificates, the key CertSign flag must be on.

SubjectAltName
Checked for form only. Must be marked critical if the subject DN is empty.

SubjectKeyIdentifier
Checked for form only.

All other extensions are ignored if they are not marked critical. Unsupported critical extensions prevent certificate validation.

CRL extensions and CRL entry extensions

PKITP supports the following CRL and CRL entry extensions, which are checked for form only:

CRL extensions:

- AuthorityKeyIdentifier

- CRLNumber
- IssuerAltName
- IssuingDistributionPoint

CRL entry extensions:

- CertificateIssuer
- CRLReason
- HoldInstructionCode
- InvalidityDate

All other extensions are ignored if they are not marked critical. Unsupported critical extensions prevent certificate validation.

Files for PKITP

The following table lists files for PKITP:

Table 100. Summary of information about important files for PKITP

File	Description	Source location (default)
`Makefile.pkitpsamp`	Makefile for `pkitpsamp.c`.	`/usr/lpp/pkiserv/samples/`
`install_pkitp`	Program that registers the PKI Services Trust Policy plug-in with OCSF.	`/usr/lpp/pkiserv/bin`
`pkitp_ivp`	This program verifies that the plug-in installed successfully.	`/usr/lpp/pkiserv/bin`
`pkitp.h`	Contains #`define` definitions for applications calling the PKI Services OCSF Trust Policy.	`/usr/lpp/pkiserv/include/`
`pkitp.so`	This is the OCSF Trust Policy plug-in for PKI Services.	`/usr/lpp/pkiserv/lib`
`pkitpsamp.c`	Sample application program (in the C language) to call the PKI Trust Policy plug-in.	`/usr/lpp/pkiserv/samples`

Configuring and getting started with PKITP

If you have not already installed and configured OCSF and OCEP, you need to do so now. Follow the instructions in "Tasks to perform before configuring PKITP" on page 35, and then perform the following post-installation instructions.

The PKITP must be registered with OCSF before being used.

Steps for configuring PKITP

Before you begin

If you have not already done so, run the OCSF and OCEP install and verification scripts.

Procedure

Perform the following steps to install and configure PKITP:

1. Run the PKITP post installation script by entering the following command:

```
/usr/lpp/pkiserv/bin/install_pkitp
```

The program prompts you for certain information. Assuming PKI Services has been installed in its default location, answer the prompts as follows:

Prompt	Response
addin directory?	`/usr/lpp/pkiserv/lib`
addin filename?	`pkitp.so`
action? [install **|** uninstall]	`install`

You know you are done and that the installation was successful when you see the following:

```
Installing IBMPKITP…
Addin successfully installed.
```

2. Update your C/C++ environment variable _CEE_RUNOPTS to include `XPLINK(ON)` if it does not already include it. For example, execute the following command from a UNIX shell.

```
export _CEE_RUNOPTS=$_CEE_RUNOPTS' XPLINK(ON)'
```

3. To verify that the installation was successful, run the verification program (`/usr/lpp/pkiserv/bin/pkitp_ivp`).

   You know you are done and that the verification program ran successfully when you see the following:

```
Starting pkitp IVP
Initializing CSSM
CSSM Initialized
Attaching pkitp
Attach successful, Detaching pkitp
Detach of pkitp successful
Completed pkitp IVP
```

Programming Interface Information

Trust Policy API

PKITP supports only one API, `CSSM_TP_PassThrough`. The globally unique identifier (GUID) for this plug-in is: `{01EBC8AC-CC6F-450c-83B4-F0BE0FBE78F9}`. (Before an application can use a module, an installation application must register the module's name, location, and description with OCSF. The name given to a module includes both a logical name and a GUID. The logical name is a string the module developer chooses to describe the module. The GUID is a structure used to differentiate between service provider modules in the OCSF registry.)

CSSM_TP_PassThrough

Purpose

This function lets applications call TP module-specific operations that have been exported. For PKITP, the module-specific operations support certificate chain validation, based on the CA and SITE certificates that are contained within a key ring.

Format

```
void * CSSMAPI CSSM_TP_PassThrough
      (CSSM_TP_HANDLE TPHandle,
      CSSM_CL_HANDLE CLHandle,
      CSSM_DL_HANDLE DLHandle,
      CSSM_DB_HANDLE DBHandle,
      CSSM_CC_HANDLE CCHandle,
      uint32 PassThroughId,
      const void *InputParams)
```

Parameters

TPHandle
Handle to this Trust Policy module (PKITP).

CLHandle
Not used. PKITP ignores this.

DLHandle
Not used. PKITP ignores this.

DBHandle
Not used. PKITP ignores this.

CCHandle
Not used. PKITP ignores this.

PassThroughId
Used to indicate the pass-through service requested. Two services are provided:

- Service 1 **CertGroupVerify** (TP_VERIFY_PASSTHROUGH)
- Service 2 **FreeEvidence** (TP_FREE_EVIDENCE_PASSTHROUGH)

InputParams
Pointer to the API-caller-provided input parameter structure. The same structure is used for both pass-through functions. It is declared in `pkitp.h` as follows:

```
typedef struct tp_verify_extra {

   /* similar parameters as TP_CertGroupVerify */
   CSSM_CL_HANDLE CLHandle;
   CSSM_DL_DB_LIST_PTR DBList;
   unsigned int reserved; //@L1C
   CSSM_TP_STOP_ON VerificationAbortOn;
   CSSM_CERTGROUP_PTR CertToBeVerified;

   /* extra parameters: input */
   TP_INITIALPOLICY_PTR InitialPolicy;
   time_t CurrentTime;
   time_t ValidationTime;

   /* extra parameters: output */
   CSSM_BOOL result;
   uint32 DLStatusCode;                    // Status code from DL failures
   uint32 DLindex;                         // Index (from 0) into DBList
   TP_EVIDENCE_PTR Evidence;

} TP_VERIFY_EXTRA, *TP_VERIFY_EXTRA_PTR;
```

The DB list

This DBList contains one or more handles to open DB stores. The last entry in this list must be a handle to an OCEPDL DB (a real or *virtual* SAF key ring). The key ring is used to declare the list of trusted CA and SITE certificates. Like the OCEP Trust Policy, certificate chains to verify must originate from one of these trusted CAs (anchors) or the end-entity certificate must be one of the SITE certificates. Also like the OCEP Trust Policy, if the security product (SAF) marks any certificate in the candidate chain NOTRUST, the certificate chain fails validation.

The other entries in the list are used for LDAPDL DB stores. PKITP runs through these to locate CRLs and intermediate CA certificates. For each item PKITP requests, the LDAPDLs are queried in the order in which they appear in the list. The search stops the first time an LDAPDL returns an item or when the OCEPDL is reached. No query is made to the OCEPDL to locate CRLs or intermediate CA certificates.

The initial policy

The following optional, caller-provided and initialized structure defines InitialPolicy. PKITP uses the default values if the structure is not provided:

```
typedef struct tp_initialpolicy {

   /* initial-policy-set */
   uint32 NumberofPolicyIdentifiers; // number of application specific
                                     // policy OIDs (defaults to 0)
   CSSM_OID_PTR PolicyIdentifiers;   // Address of array of policy OIDs
                                     // or 0
   uint32 useCRLs;                   // 0 - no CRL processing
                                     // 1 - Check CRLs only if current CRLs found
                                     // 2 - Strong CRL checking (default)
   /* initial-explicit-policy indicator */
   CSSM_BOOL initialExplicitPolicy; // If true, indicates PKITP should
                                    // consider policy set critical
                                    // defaults to false
   /* initial-policy-mapping-inhibit indicator */
   CSSM_BOOL initialPolicyMappingInhibit; // not used, ignored

} TP_INITIALPOLICY, *TP_INITIALPOLICY_PTR;
```

The evidence

The following optional, caller-provided structure defines the evidence. This structure is used to return information relative to the validation decision PKITP makes. The caller must free the data areas returned. (The **FreeEvidence** pass-through function is provided for this.)

```
typedef struct tp_evidence {

   /* valid certification path if validation succeeds */
   CSSM_CERTGROUP_PTR CompleteCertGroup;

   /* relevant CRL if validation fails */
   CSSM_DATA_PTR CRL;

   /* relevant certificate if validation fails */
   CSSM_DATA_PTR Cert;

   /* authority-constrained-policy */
   CSSM_BOOL authAnyPolicy;
   uint32 NumberofAuthCertPolicyIdentifiers;
   CSSM_OID_PTR AuthCertPolicyIdentifiers;

   /* list of policy mappings that occurred */
   uint32 NumberOfMappedPolicies;
   TP_CSSM_OID_PAIR_PTR mappedPolicies;

} TP_EVIDENCE, *TP_EVIDENCE_PTR;
```

Error codes

Table 101 on page 514 lists the error codes that are unique to PKI Services OCSF Trust Policy (PKITP).

Table 101. PKI Services OCSF Trust Policy (PKITP) error codes

Decimal value	Error description
8001	Certificate encoding error. Incorrect CertificatePolicies extension.
8002	Certificate policies violation.
8003	Incorrect certificate distinguished name chaining.
8004	Certificate encoding error. Subject name missing.

Table 101. PKI Services OCSF Trust Policy (PKITP) error codes (continued)

Decimal value	Error description
8006	Incorrect certificate BasicConstraints extension - cA flag off in signing certificate.
8008	Incorrect certificate KeyUsage extension - keyCertSign flag off in signing certificate.
8010	Unsupported AltName form in certificate.
8013	Certificate or CRL encoding error. Signature algorithm mismatch.
8014	Certificate encoding error. Incorrect version.
8015	CRL encoding error. Incorrect version.
8016	Unsupported critical extension in certificate.
8017	Unsupported critical extension in CRL.
8018	Unsupported critical entry extension in CRL.
8019	Certificate encoding error. Duplicate extension.
8020	CRL encoding error. Duplicate extension.
8021	Certificate signature failed verification.
8022	CRL signature failed verification.
8023	Incorrect date range in certificate or CRL. NotAfter earlier than NotBefore.
8024	Certificate's date range is in the future.
8025	Certificate has expired.
8026	CRL's date range is in the future.
8027	CRL has expired.
8028	DBList incorrect, no LDAPDL DBs or non-LDAPDL specified.
8029	CRL not found.
8030	Certificate is revoked.
8031	Unable to build certificate chain.
8033	Certificate not trusted.
8034	Incorrect CRLDistributionPoints extension in certificate.
8501	Unexpected status code returned from accessing LDAPDL.
8502	Unexpected status code returned from accessing OCEPDL.
8503	DBList incorrect, no OCEPDL DB or DB empty.

Building the sample application to invoke the certificate validation service

To perform certificate validation, your server application calls the `CSSM_TP_PassThrough` API (see "CSSM_TP_PassThrough" on page 512), passing it the certificate chain to verify. The API returns a Boolean value indicating success or failure, along with additional information about the certificate chain. The `pkitpsamp.c` code sample that follows at "Code sample of the PKITP program (pkitpsamp.c)" on page 517 is provided as an aid for developing your own server application. By default, you can find this file in the `/usr/lpp/pkiserv/samples` directory.

Steps for building the sample application

Perform the following steps to build the sample application:

1. Copy the `pkitpsamp.c` program and `Makefile.pkitpsamp` to the current directory by entering the following commands:

   ```
   cp /usr/lpp/pkiserv/samples/pkitpsamp.c pkitpsamp.c
   cp /usr/lpp/pkiserv/samples/Makefile.pkitpsamp Makefile
   ```

   ____

2. Before compiling `pkitpsamp.c`, you need to edit some data (for example, information about how you want the Trust Policy to operate and where your LDAP is located). In the `pkitpsamp.c` code (see "Code sample of the PKITP program (pkitpsamp.c)" on page 517), find the section that begins with a block comment that says `// Start of application specific options`. Update the code as necessary up to the block comment that says `// End of application specific options`:

   a. If the number of LDAP servers is not 1, change `NUM_LDAPS`.

   b. Update `ldap_info` by specifying your LDAP server and port (`*myldap.mycompany*.com:389` in the sample program). If you have more than one LDAP server, you need to provide this information for each LDAP server.

   c. Replace the `"@USERID@/@KEYRINGNAME@"` default value for the `char ringname[ ]` variable in the code sample. Specify either the name of the real SAF key ring containing your trusted CA or site certificates, or the name of the *virtual* key ring that points to all your trusted CA or site certificates.

      - If using a real SAF key ring, specify the owning user ID and ring name of the real SAF key ring. **Example:** `patelusr/ring01`
      - If using a virtual key ring, replace the default value with either `*AUTH*/*` or `*SITE*/*` to point to all your trusted CA or site certificates, respectively. (The name of a virtual key ring is always an asterisk.)

   d. If necessary, change the value of `useCRLS`:

      `0`
      This means using no CRL processing. (You must specify `0` if you have no LDAP servers.)

      `1`
      This means querying LDAP for CRLs and processing those found. This is the value in the sample.

      `2`
      This means using strong CRL checking. (With strong CRL checking, a valid CRL must be found for each CA certificate in the chain.)

   e. If necessary, change `NUM_POLICIES`, the policies that the application calling PKITP uses. In the sample, this is 2. For each policy, specify the DER-encoded policy information.

   f. If necessary, change `INITIALExplicitPolicy` from the default of `FALSE` to `TRUE` if you want PKITP to require all certificates in the chain to have at least one `policydata` in the preceding list.

   ____

3. Compile and link to produce the executable, pkitpsamp, by entering the following command:

   ```
   make
   ```

   ____

4. Export LIBPATH to include `/usr/lpp/pkiserv/lib`.

   Example:

   ```
   export LIBPATH=$LIBPATH:/usr/lpp/pkiserv/lib
   ```

   ____

5. Enable program control by setting the extended attribute for pkitpsamp.

   Example:

   ```
   extattr +p pkitpsamp
   ```

Restriction: To execute the **extattr** command with the **+p** option, you must have at least READ access to the BPX.FILEATTR.PROGCTL resource in FACILITY class.

---

6. Update your C/C++ environment variable `_CEE_RUNOPTS` to include `XPLINK(ON)` if it does not already include it. For example, execute the following command from a UNIX shell.

   Example:

   ```
   export _CEE_RUNOPTS=$_CEE_RUNOPTS' XPLINK(ON)'
   ```

---

7. Run the `pkitpsamp.c` in your own directory by entering the following command:

   ```
   pkitpsamp
   ```

Code sample of the PKITP program (pkitpsamp.c)

Note: The following listing might not be identical to the code sample shipped with the product. For the most current sample, see the `/usr/lpp/pkiserv/samples` directory.

```
/**********************************************************************/
/* This file contains sample code. IBM provides this code on an      */
/* 'as is' basis without warranty of any kind, either express or     */
/* implied, including but not limited to, the implied warranties     */
/* of merchantability or fitness for a particular purpose.           */
/**********************************************************************/
/**********************************************************************/
/*                                                                    */
/*    Licensed Materials - Property of IBM                            */
/*    5650-ZOS                                                        */
/*    Copyright IBM Corp. 2001, 2019                                  */
/*    Status = HKY77C0                                                */
/*                                                                    */
/**********************************************************************/
/*    Sample use of IBM PKITP program                                 */
/*                                                                    */
/*    Purpose: Program attaches needed CSSM modules, then prompts     */
/*             the user for filename(s) containing DER encoded        */
/*             certificates.  The certificate(s) are read from the    */
/*             file, then passed to PKITP for verification.           */
/*             A summary of the results are printed to stdout.        */
/*                                                                    */
/* Caution: In order to run this sample program, modification MUST    */
/*          BE MADE to several values assigned to the following       */
/*          variables that are defined between the block comment      */
/*          containing the text "Start of application specific       */
/*          options" and the block comment containing the text       */
/*          "End of application specific options"(without the        */
/*          quotation marks):                                         */
/*                                                                    */
/*   #define NUM_LDAPS 1                                              */
/*           Define the number of LDAP servers that PKITP should      */
/*           query for certificates, CRLs and ARLs.  This can be 0,   */
/*           if entire certificate chain will be passed as input to   */
/*           PKITP AND caller requests to NOT process CRLs/ARLs (see  */
/*           useCRLs option below).                                   */
/*                                                                    */
/*   struct ldap_info ldapserver[NUM_LDAPS] =                         */
/*                     { "@LDAPSERVERNAME:PORTNUMBER@",              */
/*                       "@LDAPUSER@",                                */
/*                       "@LDAPUSERPASSWORD@"};                       */
/*           If NUM_LDAPS > 0, then ldapserver array should define    */
/*           the LDAP server:port, user and password for each LDAP    */
/*           server. Replace @LDAPSERVERNAME:PORTNUMBER@ with the     */
/*           appropriate ldap server name and port number (e.g        */
/*           myldap.mycompany.com:389 ). Replace @LDAPUSER@ with the */
/*           appropriate ldap admin user name (e.g cn=root) and       */
/*           @LDAPUSERPASSWORD@ with the password for the specified   */
/*           ldap user name (e.g rootpw)                              */
/*                                                                    */
/*   char keyring[] = "@USERID@/@KEYRINGNAME@";                       */
/*           Define the SAF keyring containing trusted CA and/or      */
/*           site certificates. Format is "USERID/keyname". Replace   */
/*           @USERID@ with the userid of the keyring owner and        */
/*           @KEYRINGNAME@ with the name of the keyring. (e.g         */
/*           IBMUSER/CAring) Note that the userid and the keyring     */
/*           names are case sensitive so the userid is all            */
```

```
/*           uppercase and the keyring name is mixed case in this    */
/*           example.                                                */
/*                                                                   */
/*   #define USECRLS 1                                               */
/*           Define how the useCRLs option should be set.            */
/*             Set to 0 if no CRL processing is to be performed      */
/*             Set to 1, if LDAP is to be queried for CRLs and       */
/*             process the CRLs found.                               */
/*             Set to 2, for strong CRL checking (With strong CRL    */
/*             cheching, a valid CRL must be found for each CA       */
/*             certificate in the chain.)                            */
/*                                                                   */
/*   #define NUM_POLICIES 2                                          */
/*   static unsigned char my_policy1[5] =                            */
/*          {0x06,0x03,0x2a,0x03,0x04};       // DER encoded 2.3.4   */
/*   static unsigned char my_policy2[7] =                            */
/*          {0x06,0x05,0x2a,0x03,0x03,0x02,0x01}; // DER 2.3.3.2.1   */
/*   CSSM_DATA policydata[NUM_POLICIES] =                            */
/*          {{sizeof(my_policy1),(unsigned char *)my_policy1},       */
/*           {sizeof(my_policy2),(unsigned char *)my_policy2}};      */
/*           Define the policies that the application calling PKITP  */
/*           uses. These become important if a certificate in the    */
/*           certificate chain has a critically marked policy        */
/*           extension.  At least one policy that is listed in such  */
/*           a critically marked policy extension, must appear in    */
/*           the list defined here or PKITP will return certificate  */
/*           policy error.                                           */
/*                                                                   */
/*   #define INITIALExplicitPolicy FALSE                             */
/*           Set to true if you want PKITP to require that all       */
/*           certificates in chain to have at least one policy       */
/*           listed by the policydata defined above.                 */
/*                                                                   */
/*******************************************************************/
#pragma runopts("XPLINK(ON)")
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <cssm.h>
#include <ibmocepdl.h>
#include <cssmapi.h>
#include <cssmtype.h>
#include <pkitp.h>
#include <ldapdl.h>

struct ldap_info
 {
 char * ldapserver;
 char * ldapauthuser;
 char * ldapauthpass;
 };

//-----------------------------------------------------------
// storage function definitions needed to talk to CSSM
//-----------------------------------------------------------

#ifdef __cplusplus
extern "C"
#endif
void * OurMalloc(size_t size, void * allocRef)
{
   return malloc(size);
}

#ifdef __cplusplus
extern "C"
#endif
void OurFree(void* memPtr, void * allocRef)
{
   free(memPtr);
}

#ifdef __cplusplus
extern "C"
#endif
void * OurRealloc(void * memPtr,
                      size_t size,
                      void * allocRef)
{
   return realloc(memPtr, size);
}

#ifdef __cplusplus
extern "C"
#endif
void * OurCalloc(size_t num,
              size_t size,
              void* allocRef)
{
   return calloc(num, size);
}

static CSSM_API_MEMORY_FUNCS memoryFuncs; // used to pass function addresses to CSSM
```

```
//-----------------------------------------------------------
// internal function declarations
//-----------------------------------------------------------
int connectTP(char * ringname,
              int number_ldap,
              CSSM_DL_DB_LIST *,
              CSSM_TP_HANDLE *);  //@D1C

void disconnectTP(CSSM_DL_DB_LIST *, CSSM_TP_HANDLE);

int buildCertGroup(CSSM_CERTGROUP *, char * [], uint32);
void verifyCertGroup(CSSM_CERTGROUP certgroup,
                     CSSM_DL_DB_LIST * datasources_ptr,
                     CSSM_TP_HANDLE tphandle);

void
reportCertGroupVerify
   (TP_VERIFY_EXTRA extraVerifyInfo);

void printEvidence(TP_EVIDENCE_PTR evidence_ptr);

void freeCertGroup(CSSM_CERTGROUP * certGroupPtr);
///////////////////////////////////////////////////////////////////////////
//
// Start of application specific options
//
// The defines and declarations that follow should be altered to fit the
// particular application calling PKITP.
//
///////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////
// Define the number of LDAP servers that PKITP should query for certificates,
// CRLs and ARLs.  This can be 0, if entire certificate chain will be passed as
// input to PKITP AND caller requests to NOT process CRLs/ARLs (see useCRLs
// option below).
//
// If NUM_LDAPS > 0, then ldapserver array should define the LDAP server:port,
// user and password for each LDAP server, as this example shows.
/////////////////////////////////////////////////////////////////////////////
#define NUM_LDAPS 1

#if NUM_LDAPS != 0   //@D1A
struct ldap_info ldapserver[NUM_LDAPS] =
                                  { "@LDAPSERVERNAME:PORTNUMBER@",  // LDAP server:port
                                    "@LDAPUSER@",                   // user
                                    "@LDAPUSERPASSWORD@"};          // password
#endif  //@D1A

/////////////////////////////////////////////////////////////////////////////
// Define the SAF keyring containing trusted CA and/or site certificates.
// Format is "USERID/keyname"
/////////////////////////////////////////////////////////////////////////////
char keyring[] = "@USERID@/@KEYRINGNAME@";

/////////////////////////////////////////////////////////////////////////////
// Define how the useCRLs option should be set.
// Set to 0 if no CRL processing to be done
// Set to 1, if we are to query LDAP for CRLs and process those found
// Set to 2, for strong CRL checking -- must find CRLs in LDAP.
/////////////////////////////////////////////////////////////////////////////

#define USECRLS 1

/////////////////////////////////////////////////////////////////////////////
// Define the policies that the application calling PKITP uses.
//
// These become important if a certificate in the certificate chain has a
// critically marked policy extension.  At least one policy
// that is listed in such a critically marked policy extension, must appear
// in the list defined here or PKITP will return certificate policy error.
/////////////////////////////////////////////////////////////////////////////

#define NUM_POLICIES 2

#if NUM_POLICIES != 0   //@D1A
static unsigned char my_policy1[5] = {0x06,0x03,0x2a,0x03,0x04};  // DER encoded 2.3.4
static unsigned char my_policy2[7] = {0x06,0x05,0x2a,0x03,0x03,0x02,0x01};  // DER 2.3.3.2.1

CSSM_DATA policydata[NUM_POLICIES] = {{sizeof(my_policy1),(unsigned char *)my_policy1},
                                      {sizeof(my_policy2),(unsigned char *)my_policy2}};
#endif   //@D1A


#define INITIALExplicitPolicy FALSE  // Set to true if you want PKITP to require that all
                                     // certificates in chain have at least one policy
                                     // listed by our policydata defined above

///////////////////////////////////////////////////////////////////////////
//
// End of application specific options
//
```

```
//////////////////// ////////////////////////////////////////////////////

//------------------------------------------------------------
// main
//------------------------------------------------------------
int
main(int argc, char* argv[])
{

 CSSM_DL_DB_LIST datasources;
 CSSM_TP_HANDLE  tphandle = 0;
 CSSM_CERTGROUP certGroup;
 int repeating = 1;
 char buffer[1024];
 int num_certs = 0;
 char * cert_files[25];
 char * next_file;
 char * input;

 int rc;

 rc = connectTP(keyring,NUM_LDAPS, &datasources, &tphandle); //@D1C
 if (rc == 0)
  {
  ////////////////////////////////////////////////////////////////////////////
  // prompt for certificates to verify
  ///////////////////////////////////////////////////////////////////////////
  do
    {
    num_certs = 0;
    printf("Enter filename(s) of certificate(s).  (List EE first).  ");
    printf("Blank line to quit.\n");

    if ((input = gets(buffer)) != NULL)      // get input line
      {
      next_file = strtok(input," ");
      while ((next_file != NULL) && (num_certs < 25)) // tokenize it
        {
        cert_files[num_certs] = next_file;
        num_certs++;
        next_file = strtok(NULL," ");
        }
      }

    ///////////////////////////////////////////////////////////////////////
    // If we were given a list of files containing certificates, input them to TP
    ////////////////////////////////////////////////////////////////////////
    if (num_certs > 0)
      {
      rc = buildCertGroup(&certGroup, cert_files, num_certs);
      if (rc == 0)
        {
        verifyCertGroup(certGroup, &datasources, tphandle);
        freeCertGroup(&certGroup);
        }
      }
    } while (num_certs > 0);
  }
 disconnectTP(&datasources, tphandle);
}

//------------------------------------------------------------
// connectTP
//
// Purpose:  connect to the datasources PKITP needs
// then connect to the PKITP
//
// Input:  ringname - string containing "USERID/ringname" of SAF
//                    keyring containing trusted CA and/or SITE certificates
//         number_ldap - number of ldap servers
//         ldapserver - array of ldap_info structures
//
// Output: The CSSM_DL_DB_LIST structure addressed by datasources will have
//         been initialized with the various handles that CSSM_ModuleAttach
//         and CSSM_DL_DbOpen calls have returned
//         The CSSM_TP_HANDLE addressed by tphandle_ptr will have been initialied.
//         int returned will be 0 if successful, -1 if not successful.
//------------------------------------------------------------

int connectTP(char * ringname,
              int number_ldap,
              CSSM_DL_DB_LIST * datasources_ptr,
              CSSM_TP_HANDLE * tphandle_ptr)        //@D1C
  {
   uint32 status = 0;
   int z;
   CSSM_VERSION cssm_version = {CSSM_MAJOR, CSSM_MINOR};
   CSSM_DB_ACCESS_TYPE access = { CSSM_TRUE,
                                  CSSM_FALSE,
                                  CSSM_FALSE,
                                  CSSM_FALSE};
   CSSM_VERSION DL_version;
   CSSM_DL_HANDLE LDAP_dlhandle;
```

```
    CSSM_MODULE_INFO* moduleInfoPtr;
    void * voidptr;
    CSSM_DB_ACCESS_TYPE accessRequest = { CSSM_TRUE,     // ReadAccess
                                          CSSM_TRUE,     // WriteAccess
                                          CSSM_FALSE,    // PrivilegedMode
                                          CSSM_FALSE  }; // Asynchronous


    memoryFuncs.malloc_func = OurMalloc;
    memoryFuncs.free_func = OurFree;
    memoryFuncs.realloc_func = OurRealloc;
    memoryFuncs.calloc_func = OurCalloc;
    memoryFuncs.AllocRef = NULL;

    DL_version.Major = IBMOCEPDL_MAJOR_VERSION;
    DL_version.Minor = IBMOCEPDL_MINOR_VERSION;

    datasources_ptr->NumHandles = number_ldap + 1;
    voidptr = malloc(sizeof(CSSM_DL_DB_HANDLE)*(number_ldap +1));   // get storage for DBlist
    if (voidptr == NULL) {                                       // @D3A
       printf("connectTP unable to obtain memory: line %d\n",__LINE__); // @D3A
       return -1;                                                // @D3A
    }                                                            // @D3A

    memset(voidptr,0,(sizeof(CSSM_DL_DB_HANDLE)*(number_ldap +1))); // zero it
    datasources_ptr->DLDBHandle = (CSSM_DL_DB_HANDLE *)voidptr;

    if (CSSM_Init(&cssm_version, &memoryFuncs, NULL) != CSSM_OK)
         {
         printf("Failed CSSM_Init: %d, line %d\n",CSSM_GetError()->error,__LINE__);
         return -1;
         }

    ////////////////////////////////////////////////////////////////////////////////
    // attach to LDAP and open each LDAP DB
    ////////////////////////////////////////////////////////////////////////////////
 #if NUM_LDAPS != 0   //@D1A
    if (number_ldap > 0)              // if we have any LDAP sources
      {
      moduleInfoPtr = CSSM_GetModuleInfo((CSSM_GUID*)&LDAPDL_GUID,
                                          CSSM_SERVICE_DL,
                                          CSSM_ALL_SUBSERVICES,
                                          CSSM_INFO_LEVEL_ALL_ATTR);
      if (!moduleInfoPtr)
        {
        printf("Failed CSSM_GetModduleInfo: %d, line %d\n",CSSM_GetError()->error,__LINE__);
        return -1;
        }

      LDAP_dlhandle = CSSM_ModuleAttach((CSSM_GUID*)&LDAPDL_GUID,
                                    &moduleInfoPtr->Version,
                                    &memoryFuncs,
                                    0,
                                    0,
                                    0,
                                    NULL,
                                    NULL);
      if (!LDAP_dlhandle)
        {
        printf("Failed CSSM_ModuleAttach: %d, line %d\n",CSSM_GetError()->error,__LINE__);
        return -1;
        }

      // connect to multiple database instances

      //-----------------------------------------------------------
      // fill in LDAP DL authentication information:
      // necessary only if user is supplying a name and password
      //-----------------------------------------------------------
      for (z = 0; z < number_ldap; z++)             // for each LDAP source
        {
        LDAP_BIND_PARMS bindParms;
        CSSM_USER_AUTHENTICATION userAuthentication = {0,0};
        CSSM_DATA userCredential = {0,0};
        CSSM_USER_AUTHENTICATION_PTR userAuthenticationPtr = 0;

        datasources_ptr->DLDBHandle[z].DLHandle = LDAP_dlhandle;
        if (ldapserver[z].ldapauthuser && ldapserver[z].ldapauthpass)   //@D1C
          {
          //-----------------------------------------------------------
          // fill in LDAP DL specific data structure: LDAP_BIND_PARMS
          //-----------------------------------------------------------
          bindParms.DN = ldapserver[z].ldapauthuser;                    //@D1C
          bindParms.SASL = 0;
          bindParms.credentials.Data = (uint8 *)ldapserver[z].ldapauthpass;  //@D1C
          bindParms.credentials.Length = strlen(ldapserver[z].ldapauthpass)+1;  //@D1C
          userCredential.Length = sizeof(LDAP_BIND_PARMS);
          userCredential.Data   = (unsigned char*)&bindParms;
          userAuthentication.Credential = &userCredential;
          userAuthenticationPtr = &userAuthentication;
          }

       //-----------------------------------------------------------
       // Open LDAP DL Database
       //-----------------------------------------------------------
       datasources_ptr->DLDBHandle[z].DBHandle = CSSM_DL_DbOpen(LDAP_dlhandle,
                                         ldapserver[z].ldapserver,             //@D1C
                                         &accessRequest,
                                         userAuthenticationPtr,
                                         (void *)0);
       if (!datasources_ptr->DLDBHandle[z].DBHandle)
         {
         printf("Failed CSSM_DL_DbOpen %d, line %d\n", CSSM_GetError()->error,__LINE__);
         return -1;
         }

      }                             // end of for each each LDAP source

      if (CSSM_FreeModuleInfo(moduleInfoPtr) == CSSM_FAIL)
      {
      printf("Failed CSSM_FreeModuleInfo, line %d, error %d\n",__LINE__,
             CSSM_GetError()->error);
      // This is not a catastrophic error, we'll continue
      }
    }                               // end if we have any LDAP sources
 #endif   //@D1A

    /////////////////////////////////////////////////////////////////////////
    // Attach to OCEP DL (to access RACF keyring)
    ///////////////////////////////////////////////////////////////////////

    datasources_ptr->DLDBHandle[number_ldap].DLHandle =
                                    CSSM_ModuleAttach(&IBMOCEPDL_GUID,
                                    &DL_version,
                                    &memoryFuncs,
                                    0,
                                    0,
                                    0,
                                    NULL,
                                    NULL);


    if (!(datasources_ptr->DLDBHandle[number_ldap].DLHandle))
      {
      printf("Failed CSSM_ModuleAttach: %d, line %d\n",CSSM_GetError()->error,__LINE__);
      return -1;
      }

    datasources_ptr->DLDBHandle[number_ldap].DBHandle =
                 CSSM_DL_DbOpen(datasources_ptr->DLDBHandle[number_ldap].DLHandle,
                                ringname,
                                &access,
                                NULL,
                                NULL);

    if (!(datasources_ptr->DLDBHandle[number_ldap].DBHandle))
      {
      printf("Failed CSSM_DL_DbOpen %d, line %d\n", CSSM_GetError()->error,__LINE__);
      return -1;
      }


    ///////////////////////////////////////////////////////////////////////
    // Attach to PKITP
    ///////////////////////////////////////////////////////////////////////
    moduleInfoPtr = CSSM_GetModuleInfo((CSSM_GUID*)&PKITP_GUID,
                                       CSSM_SERVICE_TP,
                                       CSSM_ALL_SUBSERVICES,
                                       CSSM_INFO_LEVEL_ALL_ATTR);
    if (!moduleInfoPtr)
      {
      printf("Failed CSSM_GetModduleInfo: %d, line %d\n",CSSM_GetError()->error,__LINE__);
      return -1;
      }

    *(tphandle_ptr) = CSSM_ModuleAttach((CSSM_GUID*)&PKITP_GUID,
                                  &moduleInfoPtr->Version,
                                  &memoryFuncs,
                                  0,
                                  0,
                                  0,
                                  NULL,
                                  NULL);
    if (!(*tphandle_ptr))
      {
      printf("Failed CSSM_ModuleAttach: %d, line %d\n",CSSM_GetError()->error,__LINE__);
      return -1;
      }

    if (CSSM_FreeModuleInfo(moduleInfoPtr) == CSSM_FAIL)
       {
       printf("Failed CSSM_FreeModuleInfo, line %d, error %d\n",__LINE__,
             CSSM_GetError()->error);
       // This is not a catastrophic error, we'll continue
      }
```

```
  return 0;
}

//-------------------------------------------------------------
// disconnectTP
//
// Purpose:  to close any open databases and detach any CSSM modules
//           that connectTP attached
//
// Input:  The CSSM_DL_DB_LIST structure, CSSM_TP_HANDLE,
//         that were initialized by connectTP.
//
// Output: None
//-------------------------------------------------------------

void disconnectTP(CSSM_DL_DB_LIST * datasources_ptr, CSSM_TP_HANDLE tphandle)
  {
  int x;
  int status;

  /////////////////////////////////////////////////////////////////
  // Sever ties to LDAP
  // For each LDAP database opened -- call CSSM_DL_DbClose
  /////////////////////////////////////////////////////////////////

#if NUM_LDAPS != 0   //@D1A
  for (x = 0; x < datasources_ptr->NumHandles - 1; x++)
     {
      // we close each ldap database separately
      if (datasources_ptr->DLDBHandle[x].DBHandle)        // if we opened database
        {
        status = CSSM_DL_DbClose(datasources_ptr->DLDBHandle[x]);
        if (status != 0)
          {
          printf("Failed CSSM_DL_DbClose %d, line %d\n", CSSM_GetError()->error,__LINE__);
          // we continue trying to close other stuff
          }
        }
     }

   /////////////////////////////////////////////////////////////////
   // Now detach the LDAP module
   /////////////////////////////////////////////////////////////////
   if (datasources_ptr->DLDBHandle[0].DLHandle)
     {
     if ((status = CSSM_ModuleDetach(datasources_ptr->DLDBHandle[0].DLHandle)) != 0)
       {
       printf("Failed CSSM_ModuleDetach: %d, line %d\n", CSSM_GetError()->error,__LINE__);
       // we continue trying to close other stuff
       }
     datasources_ptr->DLDBHandle[0].DLHandle = 0;   // clear handle
     }
#endif  //@D1A

  /////////////////////////////////////////////////////////////////
  // Say goodbye to OCEP
  /////////////////////////////////////////////////////////////////
   status = CSSM_DL_DbClose(datasources_ptr->DLDBHandle[datasources_ptr->NumHandles - 1]);
   if (status != 0)
      {
      printf("Failed CSSM_DL_DbClose %d, line %d\n", CSSM_GetError()->error,__LINE__);
      // we continue trying to close other stuff
      }

   if (datasources_ptr->DLDBHandle[datasources_ptr->NumHandles - 1].DLHandle)
     {
     if ((status = CSSM_ModuleDetach(datasources_ptr->DLDBHandle[datasources_ptr->NumHandles - 1].DLHandle)) !
= 0)
       {
       printf("Failed CSSM_ModuleDetach: %d, line %d\n", CSSM_GetError()->error,__LINE__);
       // we continue trying to close other stuff
       }
     datasources_ptr->DLDBHandle[datasources_ptr->NumHandles - 1].DLHandle = 0;
     }

  /////////////////////////////////////////////////////////////////
  // Farewell PKITP
  /////////////////////////////////////////////////////////////////
  if (tphandle)
    {
    if ((status = CSSM_ModuleDetach(tphandle)) != 0)
      {
      printf("Failed CSSM_ModuleDetach: %d, line %d\n", CSSM_GetError()->error,__LINE__);
      // we continue trying to close other stuff
      }
    }

 return;
}

 /**********************************************************************
  *  name: buildCertGroup - read certificates from files, set up
  *        CSSM_CERTGROUP to reference input certificates
  *
  *  input: CSSM_CERTGROUP * -- addresses unintialized CSSM_CERTGROUP
  *         certFile - array of strings containing names of files that
  *             have DER encoded certificates to be verified by PKITP
  *         certCount - number of elements (strings) in certFile
  *
  *  output: returns CSSM_OK if all certificates read
  *          -  CSSM_CERTGROUP will have NumCerts set and CertList
  *             will be the address of array of certificates
  *          returns CSSM_FALSE if error reading a file
  ***********************************************************************/

 int buildCertGroup(CSSM_CERTGROUP * certGroupPtr,
                    char * certFile[], uint32 certCount)
 {
   FILE      * inFile;
   CSSM_DATA * certArray = (CSSM_DATA *) calloc(certCount,sizeof(CSSM_DATA));
   uint32    i, certSize;

   if (certArray == NULL) // If calloc failed, exit now
     return(CSSM_FAIL);                                       // @D3A

   certGroupPtr->NumCerts = certCount;
   certGroupPtr->CertList = certArray;

   for (i=0; i < certCount; i++) {
     inFile = fopen(certFile[i],"rb");
     if (!inFile) {
       printf("File %s could not be opened\n",certFile[i]);
       if (i > 1)          // if we've read any certs before this
       {
         certGroupPtr->NumCerts = i - 1;   // indicate how many read
         freeCertGroup(certGroupPtr);      // free alloc'd storage
       }
       return(CSSM_FAIL);
     }
     /* Find size of certificate file */
     fseek(inFile,0L,SEEK_END);
     certSize = ftell(inFile);
     rewind(inFile);

     /* Read in certificate data*/
     certArray[i].Length = certSize;
     certArray[i].Data   = (uint8 *)calloc(certSize, sizeof(char));
     if (certArray[i].Data == NULL) {  // If calloc failed         @D3A
       if (i > 1)          // if we've read any certs before this  @D3A
       {                                                        // @D3A
         certGroupPtr->NumCerts = i - 1;   // indicate how many read @D3A
         freeCertGroup(certGroupPtr);      // free alloc'd storage @D3A
       }                                                        // @D3A
       return(CSSM_FAIL);                                       // @D3A
     }                                                          // @D3A
     fread(certArray[i].Data, 1, certSize, inFile);
     fclose(inFile);
   }
   return(CSSM_OK);
 }


  /**********************************************************************
   *  name: verifyCertGroup - call the Trust Policy (FINALLY)
   *
   *  purpose:  call CSSM_TP_PassThrough (PKITP) to verify certificate(s)
   *            call reportCertGroupVerify (internal routine to display
   *               results to stdout
   *            call CSSM_TP_PassThrough (PKITP) to free storage related to
   *               results
   *
   *  input:    CSSM_CERTGROUP containing number of and array of certificates
   *            CSSM_DL_DB_LIST containing CSSM handles for LDAP and OCEP
   *            CSSM_TP_HANDLE  CSSM handle for PKITP
   *
   *  output:   none
   *
   **********************************************************************/

   void verifyCertGroup(CSSM_CERTGROUP certgroup,
                        CSSM_DL_DB_LIST * datasources_ptr,
                        CSSM_TP_HANDLE tphandle)
   {
   /////////////////////////////////////////////////////////////////////
   //
   // While there are only 3 parameters on CSSM_TP_PassThrough call to PKITP:
   //  - the CSSM_TP_HANDLE,
   //  - the function code "TP_VERIFY_PASSTHROUGH" and
   //  - a pointer to the TP_VERIFY_EXTRA structure.
   // TP_VERIFY_EXTRA structure contains many parameters, including the address of
   // TP_INITIALPOLICY structure that can be used to override the default
   // policy settings and the address of TP_VERIFY_EXTRA which PKITP can use
   // to pass back more detailed results.
   //////////////////////////////////////////////////////////////////////
   TP_INITIALPOLICY initialPolicyPreferences;
   TP_EVIDENCE pkixEvidence;
```

```
   TP_VERIFY_EXTRA extraVerifyInfo;

   memset(&extraVerifyInfo, 0x00, sizeof(TP_VERIFY_EXTRA));     // @D3A
   /////////////////////////////////////////////////////////////////////////
   // The field initialPolicyMappingInhibit in TP_INITIALPOLICY is not used
   // by PKITP, therefore we do not set it.
   /////////////////////////////////////////////////////////////////////////
   initialPolicyPreferences.NumberofPolicyIdentifiers = NUM_POLICIES;
#if NUM_POLICIES != 0   //@D1A
   initialPolicyPreferences.PolicyIdentifiers = policydata;
#else                   //@D1A
   initialPolicyPreferences.PolicyIdentifiers = NULL;    //@D1A
#endif                  //@D1A
   initialPolicyPreferences.initialExplicitPolicy = INITIALExplicitPolicy;
   initialPolicyPreferences.initialPolicyMappingInhibit = CSSM_FALSE;
   initialPolicyPreferences.useCRLs = USECRLS;

   /////////////////////////////////////////////////////////////////////////
   // Initialize TP_EVIDENCE fields for printEvidence in case call
   // to PKITP, is not successful.
   pkixEvidence.CompleteCertGroup = NULL;  /* @D0A */
   pkixEvidence.CRL       = NULL;          /* @D0A */
   pkixEvidence.Cert      = NULL;          /* @D0A */
   /////////////////////////////////////////////////////////////////////////

   ////////////////////////////////////////////////////////////////////////
   // The following fields in TP_VERIFY_EXTRA are not used by PKITP.
   // CLHandle, PolicyIdentifiers and NumberofPolicyIdentifiers
   // (not to be confused with fields of same name in TP_INITIALPOLICY structure),
   // AnchorCerts and NumberofAnchorCerts.
   // Therefore we do not set these fields below.
   ////////////////////////////////////////////////////////////////////////
   extraVerifyInfo.DBList                    = datasources_ptr;
   extraVerifyInfo.VerificationAbortOn       = CSSM_TP_STOP_ON_POLICY;
   extraVerifyInfo.CertToBeVerified          = &certgroup;
   extraVerifyInfo.InitialPolicy             = &initialPolicyPreferences;
   extraVerifyInfo.Evidence                  = &pkixEvidence;
   extraVerifyInfo.ValidationTime            = time(0);


   (void*)CSSM_TP_PassThrough(tphandle,
                              0,
                              0,
                              0,
                              0,
                              TP_VERIFY_PASSTHROUGH,
                              (void *)&extraVerifyInfo);

   reportCertGroupVerify(extraVerifyInfo);
   (void*)CSSM_TP_PassThrough(tphandle,
                              0,
                              0,
                              0,
                              0,
                              TP_FREE_EVIDENCE,
                              (void *)&extraVerifyInfo);
   }

 //===========================================================================
// function: reportCertGroupVerify
//===========================================================================

void
reportCertGroupVerify
   (TP_VERIFY_EXTRA extraVerifyInfo)
{
   //--------------------------------------
   // report success or failure
   //--------------------------------------
   unsigned int reported_err = CSSM_GetError()->error;

   printf("TP_VERIFY_PASSTHROUGH : ");
   if (CSSM_FALSE == extraVerifyInfo.result)
   {
    printf("FAILED.  Error code: %d\n",reported_err);
   }
   else
   {
    printf("PASSED\n");
   }

   //--------------------------------------
   // report evidence
   //--------------------------------------
   printEvidence(extraVerifyInfo.Evidence);


}

void printEvidence(TP_EVIDENCE_PTR evidence_ptr)
{
 if (evidence_ptr == NULL) return;
 if (evidence_ptr->CompleteCertGroup)
```

```
    {
    printf("CompleteCertGroup was returned containing %d certificates at address %x\n",
      evidence_ptr->CompleteCertGroup->NumCerts,
      evidence_ptr->CompleteCertGroup->CertList);
    }
  else printf("CompleteCertGroup was NULL.\n");

  if (evidence_ptr->CRL)
    {
    printf("CRL was returned of %d bytes (decimal) at address %x\n",
      evidence_ptr->CRL->Length,
      evidence_ptr->CRL->Data);
    }
  else printf("CRL was NULL.\n");

   if (evidence_ptr->Cert)
    {
    printf("Cert (failed certificate) was returned of %d bytes (decimal) at address %x\n",
      evidence_ptr->Cert->Length,
      evidence_ptr->Cert->Data);
    }
  else printf("Cert was NULL.\n");
 }

   /**********************************************************************
    *  name: freeCertGroup - Free certificate data storage              *
    **********************************************************************/

    void freeCertGroup(CSSM_CERTGROUP * certGroupPtr)
    {
    CSSM_DATA      * certArray = certGroupPtr->CertList;
    uint32           i;
    uint32           certCount = certGroupPtr->NumCerts;

    for (i=0; i <= certCount-1; i++)
     {
     free(certArray[i].Data);
     }
    free(certArray);
    return;
    }
```

End Programming Interface Information

Part 7. Troubleshooting

This part explains using logs for troubleshooting, including the following:

- Chapter 24, “Using information from SYS1.LOGREC,” on page 529 contains information about SYS1.LOGREC, which is used to record unusual runtime events, such as an exception.
- Chapter 25, “Using information from the PKI Services logs,” on page 537 contains information about using the PKI Services logs, which are ongoing, to debug problems and explains how to change logging options and display log options settings.

You can also use the iclview and vosview utilities for troubleshooting. For more information about these utilities, see “Using the iclview utility” on page 433 and “Using the vosview utility” on page 444.

Chapter 24. Using information from SYS1.LOGREC

SYS1.LOGREC keeps records of unusual runtime events, such as exceptions or unexpected return codes from calls to system services. It records hardware errors, selected software errors, and selected system conditions in the LOGREC data set. You can use the LOGREC data set as a starting point for diagnosing a problem. It supplies symptom data about the failure and shows the order in which errors occurred. After you have collected this information, you should report the problem to the IBM support center.

The following table describes the contents of the LOGREC data for PKI Services:

Table 102. LOGREC data for PKI Services

CSECT	Description
IKYP0N IKYP81 IKYP8A IKYP8B	Issued when an ABEND occurs in the one of the CSECTs running on the Monitor Thread. Primary symptom string: **Component ID (PIDS):** 5752XXPKI **Load module:** IKYPKID#L **CSECT:** IKYP0N, IKYP81, IKYP8A, or IKYP8B **Recovery routine:** ESTEXIT **Error information:** Consists of an abend code and reason code: **Abend code:** The character *S* followed by 4 hexadecimal digits or the character *U* followed by 4 decimal digits. **Reason code:** 8 hexadecimal digits.

Table 102. LOGREC data for PKI Services (continued)

CSECT	Description
IKYP8A	Issued when an exception is caught in the service thread routine IKYP8A01 or in the services thread request routine IKYP8A02. Primary symptom string: **Component ID (PIDS):** 5752XXPKI **Load module:** IKYPKID#L **CSECT:** IKYP8A **Failing routine:** IKYP8A01 or IKYP8A02 **Error information:** Consists of *either* an abend code and a reason code *or* a facility ID and a message number. **Abend code:** If present, either the character *U* followed by 4 decimal digits or the character *S* followed by 3 hexadecimal digits. **Reason code:** If present, 8 hexadecimal digits. **Facility ID:** If present, 3 characters. **Message number:** If present, 8 hexadecimal digits. Secondary symptom string: **USER** The user ID of the requestor. **FUNC** A function code of 8 hexadecimal digits.
IKYP8B	Issued when an ABEND occurs in the PC routine (or helper routines). Primary symptom string: **Component ID (PIDS):** 5752XXPKI **Load module:** IKYPKID#L **CSECT:** IKYP8B **Recovery routine:** ARREXIT **Error information:** Consists of an abend code and a reason code. **Abend code:** The character *S* followed by 4 hexadecimal digits or the character *U* followed by 4 decimal digits. **Reason code:** 8 hexadecimal digits.

Table 102. LOGREC data for PKI Services (continued)

CSECT	Description
IKYSCHDR	Issued from the dispatcher() function when an exception is caught while creating and posting a CRL to LDAP. Primary symptom string: **Component ID (PIDS):** 5752XXPKI **Load module:** IKYAPI#L **CSECT:** IKYSCHDR **Failing routine:** IKYDSPER **Error information:** Consists of *either* an abend code and a reason code *or* a facility ID and a message number. **Abend code:** If present, either the character *U* followed by 4 decimal digits or the character *S* followed by 3 hexadecimal digits. **Reason code:** If present, 8 hexadecimal digits. **Facility ID:** If present, 3 characters. **Message number:** If present, 8 hexadecimal digits. Secondary symptom string: **THREAD** The string `DISPATCHR`.

Table 102. LOGREC data for PKI Services (continued)

CSECT	Description
IKYSTART	Issued when an exception occurs during `daily_timer()` processing (general housekeeping for certificate requests and issued certificates). Primary symptom string: **Component ID (PIDS):** 5752XXPKI **Load module:** IKYAPI#L **CSECT:** IKYSTART **Failing routine:** IKYDAYTM **Error information:** Consists of *either* an abend code and a reason code *or* a facility ID and a message number. **Abend code:** If present, either the character *U* followed by 4 decimal digits or the character *S* followed by 3 hexadecimal digits. **Reason code:** If present, 8 hexadecimal digits. **Facility ID:** If present, 3 characters. **Message number:** If present, 8 hexadecimal digits. Secondary symptom string: **THREAD** The string `DAY_TIMR`.

Table 102. LOGREC data for PKI Services (continued)

CSECT	Description
IKYTIMER	Issued when an exception is caught while processing a timer event in wakeup_rtn(). Primary symptom string: **Component ID (PIDS):** 5752XXPKI **Load module:** IKYOSSRV#L **CSECT:** IKYTIMER **Failing routine:** IKYWAKUP **Error information:** Consists of *either* an abend code and a reason code *or* a facility ID and a message number. **Abend code:** If present, either the character *U* followed by 4 decimal digits or the character *S* followed by 3 hexadecimal digits. **Reason code:** If present, 8 hexadecimal digits. **Facility ID:** If present, 3 characters. **Message number:** If present, 8 hexadecimal digits. Secondary symptom string: **EVENTFUNC** The name of the event routine being processed (postEvt, createEvt, or removeEvt).

Sample LOGREC data

Figure 83 on page 534 andFigure 84 on page 535 show sample LOGREC data for PKI Services.

```
TYPE:  SYMPTOM RECORD       REPORT:  SOFTWARE EDIT REPORT           DAY YEAR
                                                        REPORT DATE: 221  01
SCP:   VS 2 REL 3                                        ERROR DATE: 221  01
                            MODEL:  9672                            HH MM SS.TH
                            SERIAL: 048288                    TIME: 19:05:16.02
SEARCH ARGUMENT ABSTRACT:
    PIDS/5752XXPKI RIDS/IKYPKID#L RIDS/IKYP8A RIDS/IKYP8A01 AB/S0C4
    FLDS/RSNCODE VALU/H00000000

SYSTEM ENVIRONMENT:
    CPU MODEL:  9672               DATE:  221  01
    CPU SERIAL: 048288             TIME:  19:05:16.02
    SYSTEM:     DCEIMGUI           BCP:   MVS
    RELEASE LEVEL OF SERVICE ROUTINE:      HBB7703
    SYSTEM DATA AT ARCHITECTURE LEVEL:     10
    COMPONENT DATA AT ARCHITECTURE LEVEL:  10
    SYSTEM DATA:  00000000 00000000                        |........|
COMPONENT INFORMATION:
    COMPONENT ID:             5752XXPKI
    COMPONENT RELEASE LEVEL:  7706
    SERVICE RELEASE LEVEL:    HKY7706
    DESCRIPTION OF FUNCTION:  PKI SERVICES DAEMON

PRIMARY SYMPTOM STRING:
    PIDS/5752XXPKI RIDS/IKYPKID#L RIDS/IKYP8A RIDS/IKYP8A01 AB/S0C4
    FLDS/RSNCODE VALU/H00000000

    SYMPTOM            SYMPTOM DATA     EXPLANATION
    ---------------    -------------    -----------
    PIDS/5752XXPKI     5752XXPKI        COMPONENT IDENTIFIER
    RIDS/IKYPKID#L     IKYPKID#L        ROUTINE IDENTIFIER
    RIDS/IKYP8A        IKYP8A           ROUTINE IDENTIFIER
    RIDS/IKYP8A01      IKYP8A01         ROUTINE IDENTIFIER
    AB/S0C4            0C4              ABEND CODE - SYSTEM
    FLDS/RSNCODE       RSNCODE          DATA FIELD NAME
    VALU/H00000000     00000000         ERROR RELATED HEXADECIMAL VALUE

SECONDARY SYMPTOM STRING:
    FLDS/USER VALU/CG422253 FLDS/FUNC VALU/H00000000

    SYMPTOM            SYMPTOM DATA     EXPLANATION

    ---------------    -------------    -----------
    FLDS/USER          USER             DATA FIELD NAME
    VALU/CG422253      G422253          ERROR RELATED CHARACTER VALUE
    FLDS/FUNC          FUNC             DATA FIELD NAME
    VALU/H00000000     00000000         ERROR RELATED HEXADECIMAL VALUE

THE SYMPTOM RECORD DOES NOT CONTAIN FREE FORMAT COMPONENT INFORMATION.
HEX DUMP OF RECORD:
  HEADER
    +000    4C831800    00000000    0001221F    19051602    |<C.............|
    +010    FF048288    96720000                            |..BHO...        |
```

Figure 83. Sample LOGREC data (part 1 of 2)

```
  SYMPTOM RECORD
     +000    E2D9F9F6    F7F2F0F4    F8F2F8F8    FFFFCA5B    |SR9672048288...$|
     +010    B64312D1    0360F103    40404040    40404040    |...J.-1.        |
     +020    4040C4C3    C5C9D4C7    E4C9F5F7    F5F2C8C2    |  DCEIMGUI5752HB|
     +030    C2F7F7F0    F3400080    00000000    00000000    |B7703 ..........|
     +040    F1F00030    00640070    005C0138    003101A0    |10.......*......|
     +050 LENGTH(0032) ==> ALL BYTES CONTAIN X'00'.
     +070    E2D9F2F1    F1F0F5F7    F5F2E7E7    D7D2C900    |SR21105752XXPKI.|
     +080    F7F7F0F6    C8D2E8F7    F7F0F640    00000000    |7706HKY7706 ....|
     +090    00000000    00000000    00000000    D7D2C940    |............PKI |
     +0A0    E28599A5    898385A2    40848185    94969540    |SERVICES DAEMON |
     +0B0    40404040    40404040    40404040    00000000    |            ....|
     +0C0    00000000    00000000    00000000    00000000    |................|
     +0D0    00000000    0B41465C    0B414668    0B414699    |.......*.......R|
     +0E0    0B4146A8    0B4146A8    0B4146A8    01000000    |...Y...Y...Y....|
     +0F0    0B4144C8    00000000    00000000    F0F1F2F3    |...H........0123|
     +100    F4F5F6F7    F8F9C1C2    C3C4C5C6    00680040    |456789ABCDEF... |
     +110    0000000F    0B414530    00000000    0B414374    |................|
     +120    00000000    F0F00000    00000008    00000008    |....00..........|
     +130    00000000    40E70030    D7C9C4E2    61F5F7F5    |.... X..PIDS/575|
     +140    F2E7E7D7    D2C940D9    C9C4E261    C9D2E8D7    |2XXPKI RIDS/IKYP|
     +150    D2C9C47B    D340D9C9    C4E261C9    D2E8D7F8    |KID#L RIDS/IKYP8|
     +160    C140D9C9    C4E261C9    D2E8D7F8    C1F0F140    |A RIDS/IKYP8A01 |
     +170    C1C261E2    F0C3F440    C6D3C4E2    61D9E2D5    |AB/S0C4 FLDS/RSN|
     +180    C3D6C4C5    40E5C1D3    E461C8F0    F0F0F0F0    |CODE VALU/H00000|
     +190    F0F0F040    0B414780    00000001    00000000    |000 ............|
     +1A0    C6D3C4E2    61E4E2C5    D940E5C1    D3E461C3    |FLDS/USER VALU/C|
     +1B0    C7F4F2F2    F2F5F340    C6D3C4E2    61C6E4D5    |G422253 FLDS/FUN|
     +1C0    C340E5C1    D3E461C8    F0F0F0F0    F0F0F0F0    |C VALU/H00000000|
     +1D0    40                                              |                |
```

Figure 84. Sample LOGREC data (part 2 of 2)

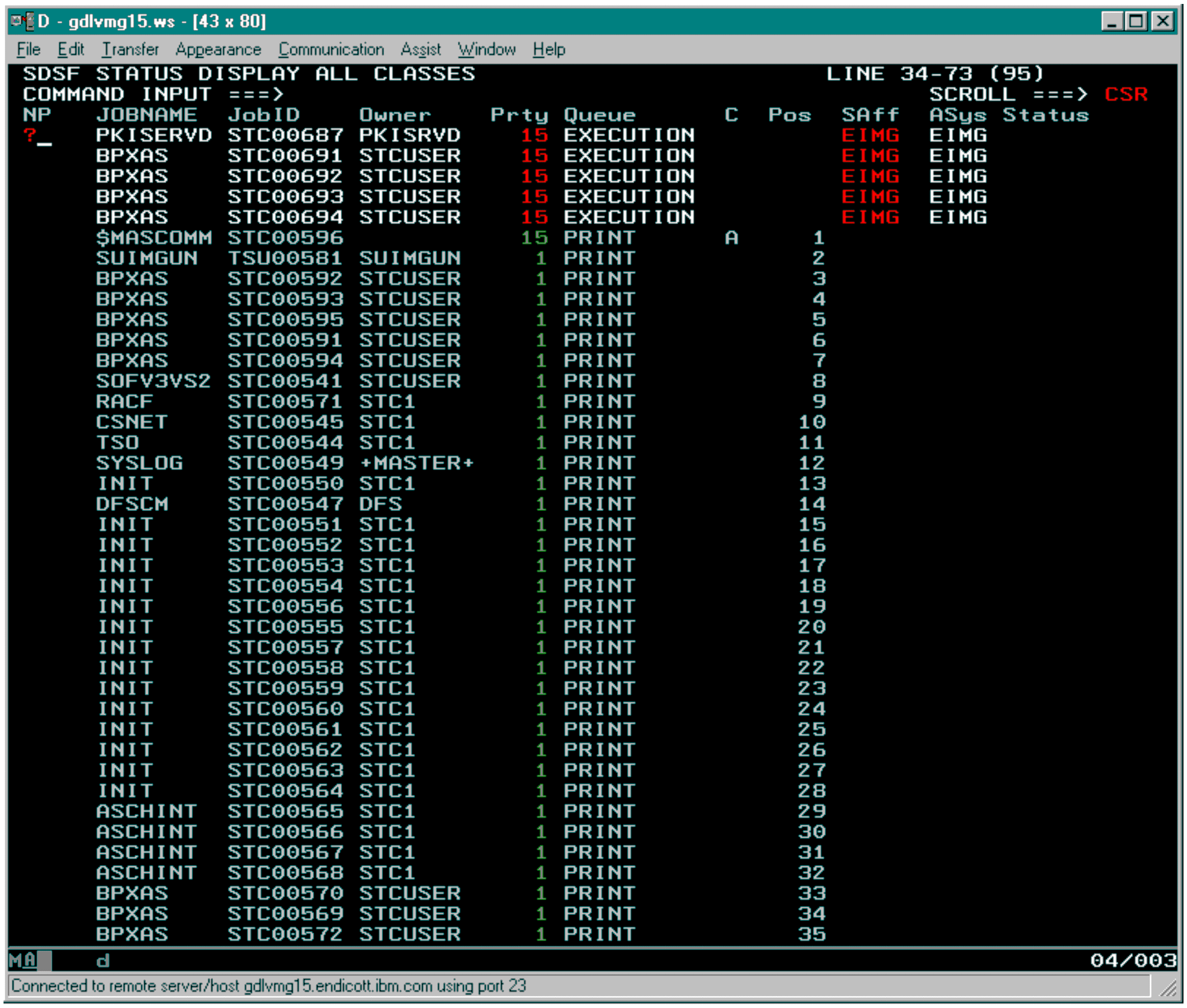
Chapter 25. Using information from the PKI Services logs

This topic explains viewing SYSOUT information. It describes the `_PKISERV_MSG_LEVEL` environment variable and lists subcomponents and message levels you can select. It explains how to display and change logging options.

Viewing SYSOUT information

To start PKI Services, you use the PKISERVD sample procedure (see "PKISERVD sample procedure to start PKI Services daemon" on page 682 for a code sample of the JCL). When you start PKI Services, error and informational messages for the PKISERVD job are written to the STDOUT and STDERR file streams. Unless you change the DD statements that specify STDOUT and STDERR in the PKISERVD sample procedure, PKI Services writes these messages to SYSOUT.

To view the SYSOUT information of a job, you use the Spool Display Search Facility (SDSF) or a comparable facility. If you are using SDSF, you can use the question mark line command (by entering a question mark in the prefix area in front of the file name) to separate the job files, including STDOUT and STDERR. Figure 85 on page 538 shows this.

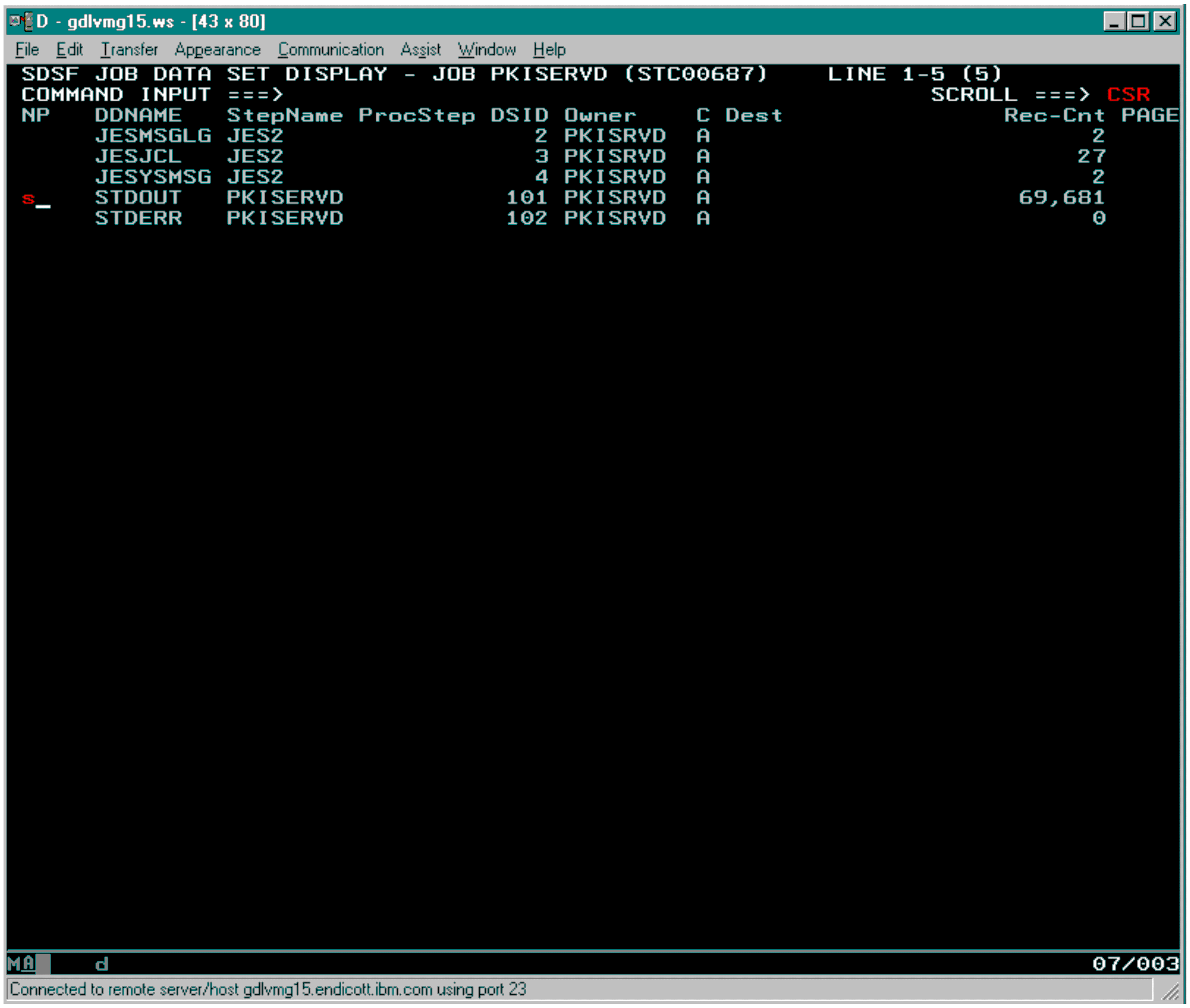

D - gdlvmg15.ws - [43 x 80]
File Edit Transfer Appearance Communication Assist Window Help

```
 SDSF STATUS DISPLAY ALL CLASSES                            LINE 34-73 (95)
 COMMAND INPUT ===>                                                SCROLL ===> CSR
 NP   JOBNAME  JobID    Owner    Prty Queue      C  Pos  SAff  ASys Status
 ?_   PKISERVD STC00687 PKISRVD   15 EXECUTION           EIMG  EIMG
      BPXAS    STC00691 STCUSER   15 EXECUTION           EIMG  EIMG
      BPXAS    STC00692 STCUSER   15 EXECUTION           EIMG  EIMG
      BPXAS    STC00693 STCUSER   15 EXECUTION           EIMG  EIMG
      BPXAS    STC00694 STCUSER   15 EXECUTION           EIMG  EIMG
      $MASCOMM STC00596           15 PRINT      A     1
      SUIMGUN  TSU00581 SUIMGUN    1 PRINT            2
      BPXAS    STC00592 STCUSER    1 PRINT            3
      BPXAS    STC00593 STCUSER    1 PRINT            4
      BPXAS    STC00595 STCUSER    1 PRINT            5
      BPXAS    STC00591 STCUSER    1 PRINT            6
      BPXAS    STC00594 STCUSER    1 PRINT            7
      SOFV3VS2 STC00541 STCUSER    1 PRINT            8
      RACF     STC00571 STC1       1 PRINT            9
      CSNET    STC00545 STC1       1 PRINT           10
      TSO      STC00544 STC1       1 PRINT           11
      SYSLOG   STC00549 +MASTER+   1 PRINT           12
      INIT     STC00550 STC1       1 PRINT           13
      DFSCM    STC00547 DFS        1 PRINT           14
      INIT     STC00551 STC1       1 PRINT           15
      INIT     STC00552 STC1       1 PRINT           16
      INIT     STC00553 STC1       1 PRINT           17
      INIT     STC00554 STC1       1 PRINT           18
      INIT     STC00556 STC1       1 PRINT           19
      INIT     STC00555 STC1       1 PRINT           20
      INIT     STC00557 STC1       1 PRINT           21
      INIT     STC00558 STC1       1 PRINT           22
      INIT     STC00559 STC1       1 PRINT           23
      INIT     STC00560 STC1       1 PRINT           24
      INIT     STC00561 STC1       1 PRINT           25
      INIT     STC00562 STC1       1 PRINT           26
      INIT     STC00563 STC1       1 PRINT           27
      INIT     STC00564 STC1       1 PRINT           28
      ASCHINT  STC00565 STC1       1 PRINT           29
      ASCHINT  STC00566 STC1       1 PRINT           30
      ASCHINT  STC00567 STC1       1 PRINT           31
      ASCHINT  STC00568 STC1       1 PRINT           32
      BPXAS    STC00570 STCUSER    1 PRINT           33
      BPXAS    STC00569 STCUSER    1 PRINT           34
      BPXAS    STC00572 STCUSER    1 PRINT           35
MA     d                                                                  04/003
```

Connected to remote server/host gdlvmg15.endicott.ibm.com using port 23

Figure 85. Separating the job files

After using the question mark line command, you can select the file you want to view by entering an *S* before this file name. Figure 86 on page 539 shows this:

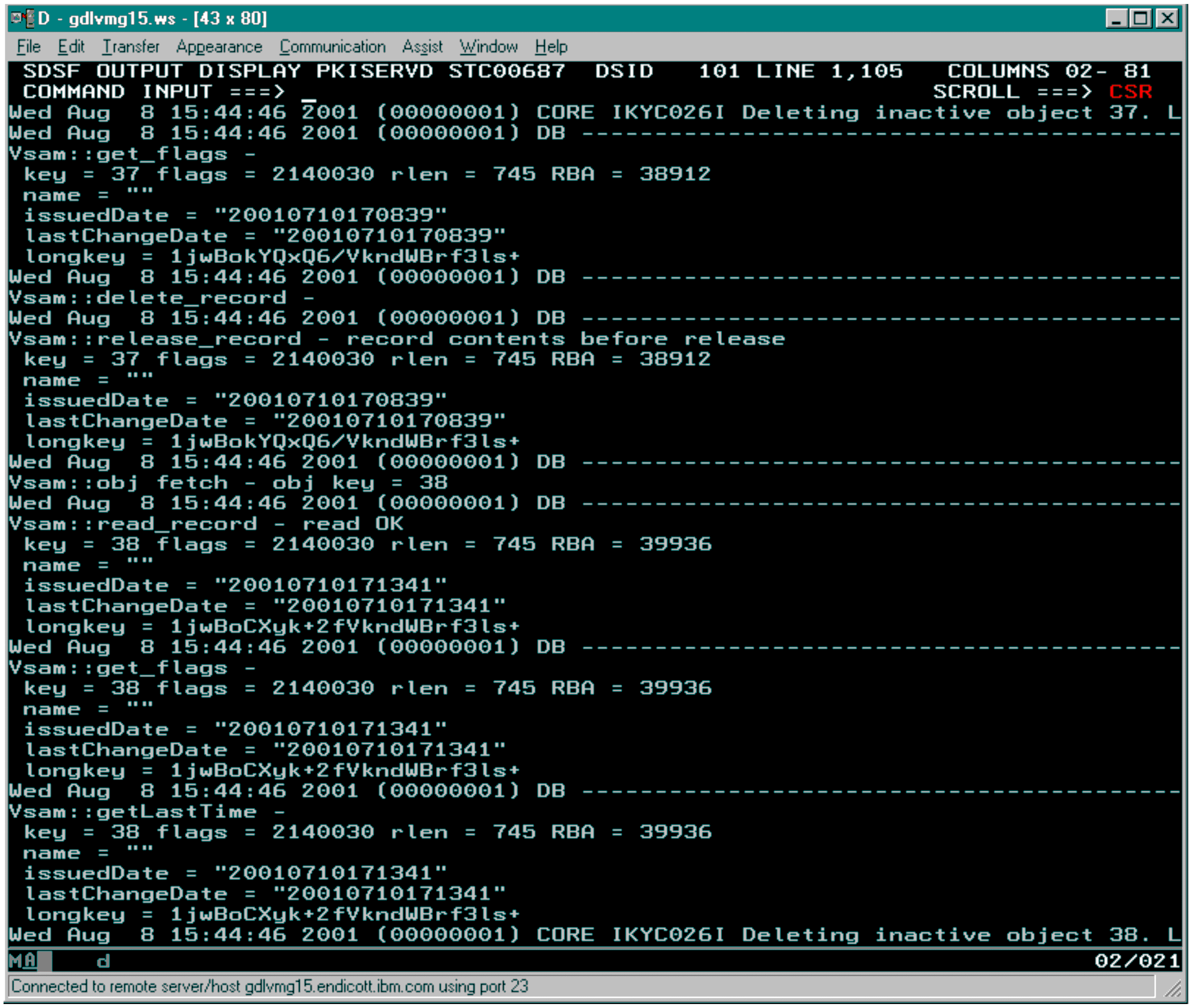

D - gdlvmg15.ws - [43 x 80]
File Edit Transfer Appearance Communication Assist Window Help

```
SDSF JOB DATA SET DISPLAY - JOB PKISERVD (STC00687)     LINE 1-5 (5)
COMMAND INPUT ===>                                          SCROLL ===> CSR
NP    DDNAME    StepName ProcStep DSID Owner    C Dest            Rec-Cnt PAGE
      JESMSGLG JES2                   2 PKISRVD  A                       2
      JESJCL   JES2                   3 PKISRVD  A                      27
      JESYSMSG JES2                   4 PKISRVD  A                       2
s_    STDOUT   PKISERVD             101 PKISRVD  A                  69,681
      STDERR   PKISERVD             102 PKISRVD  A                       0
```

```
MA      d                                                              07/003
Connected to remote server/host gdlvmg15.endicott.ibm.com using port 23
```

Figure 86. Selecting a file to view

Figure 87 on page 540 shows the messages contained in the file:


D - gdlvmg15.ws - [43 x 80]
File Edit Transfer Appearance Communication Assist Window Help

```
 SDSF OUTPUT DISPLAY PKISERVD STC00687  DSID   101 LINE 1,105    COLUMNS 02- 81
 COMMAND INPUT ===> _                                          SCROLL ===> CSR
Wed Aug  8 15:44:46 2001 (00000001) CORE IKYC026I Deleting inactive object 37. L
Wed Aug  8 15:44:46 2001 (00000001) DB -----------------------------------------
Vsam::get_flags -
 key = 37 flags = 2140030 rlen = 745 RBA = 38912
 name = ""
 issuedDate = "20010710170839"
 lastChangeDate = "20010710170839"
 longkey = 1jwBokYQxQ6/VkndWBrf3ls+
Wed Aug  8 15:44:46 2001 (00000001) DB -----------------------------------------
Vsam::delete_record -
Wed Aug  8 15:44:46 2001 (00000001) DB -----------------------------------------
Vsam::release_record - record contents before release
 key = 37 flags = 2140030 rlen = 745 RBA = 38912
 name = ""
 issuedDate = "20010710170839"
 lastChangeDate = "20010710170839"
 longkey = 1jwBokYQxQ6/VkndWBrf3ls+
Wed Aug  8 15:44:46 2001 (00000001) DB -----------------------------------------
Vsam::obj fetch - obj key = 38
Wed Aug  8 15:44:46 2001 (00000001) DB -----------------------------------------
Vsam::read_record - read OK
 key = 38 flags = 2140030 rlen = 745 RBA = 39936
 name = ""
 issuedDate = "20010710171341"
 lastChangeDate = "20010710171341"
 longkey = 1jwBoCXyk+2fVkndWBrf3ls+
Wed Aug  8 15:44:46 2001 (00000001) DB -----------------------------------------
Vsam::get_flags -
 key = 38 flags = 2140030 rlen = 745 RBA = 39936
 name = ""
 issuedDate = "20010710171341"
 lastChangeDate = "20010710171341"
 longkey = 1jwBoCXyk+2fVkndWBrf3ls+
Wed Aug  8 15:44:46 2001 (00000001) DB -----------------------------------------
Vsam::getLastTime -
 key = 38 flags = 2140030 rlen = 745 RBA = 39936
 name = ""
 issuedDate = "20010710171341"
 lastChangeDate = "20010710171341"
 longkey = 1jwBoCXyk+2fVkndWBrf3ls+
Wed Aug  8 15:44:46 2001 (00000001) CORE IKYC026I Deleting inactive object 38. L
MA      d                                                                 02/021
```

Connected to remote server/host gdlvmg15.endicott.ibm.com using port 23

Figure 87. Messages contained in the file

Notes:

1. These messages were produced when Verbose tracing was active.
2. The SYSOUT records have a logical record length of 133, so you might have to scroll to the right to see the entire record.

From left to right, each record contains:

- A time stamp
- The thread identifier, in parenthesis
- The subcomponent name (in the example that follows, this is CORE)
- The message itself, which might span multiple lines

  Informational, warning, error, and severe level messages begin with a message number. (See Chapter 26, “Messages,” on page 547.) Verbose and diagnostic level messages do not have message numbers and are not documented.

The following is an example of an informational message:

```
Wed Aug 8 15:44:46 2001 (00000001) CORE IKYC026I Deleting inactive object 37.
     Last changed at 2001/07/10 17:08:39
```

_PKISERV_MSG_LEVEL subcomponents and message levels

The _PKISERV_MSG_LEVEL environment variable specifies the subcomponent and message level for logging messages.

The following is a list of subcomponents :

Subcomponent	Meaning
*	The wildcard character (represents all subcomponents)
CORE	The core functions of PKI Services that are not specific to the other subcomponents
DB	Activity related to the object store or issued certificate list repositories
LDAP	LDAP posting operations
PKID	The PKI Services daemon address setup and infrastructure
POLICY	Certificate creation and revocation policy processing
SAF	SAF key ring, OCEP, and R_datalib calls
TPOLICY	Trust policy plug-in processing

The following is a hierarchical list of message levels:

Debug level	Meaning
S	This indicates logging only severe messages.
E	This indicates logging severe and error messages.
W	This indicates logging severe, error, and warning messages. This is the *default* message level for all subcomponents if you do not set the environment variable.
I	This indicates logging severe, error, warning, and informational messages.
D	This indicates logging severe, error, warning, informational, and diagnostic messages.
V	This indicates logging *all* messages, including verbose diagnostic messages. This is very verbose. **Guideline:** Do not use **V** level unless IBM support personnel instruct you to do so.

(For information about updating environment variables during configuration, see "Optionally updating PKI Services environment variables" on page 70.)

After PKI Services is up and running, if a problem occurs, the MVS programmer can:

- Change the logging options dynamically - by using the MODIFY (or **F**) console command
- Display the current settings - by using another MODIFY console command.

Changing logging options

To change logging options dynamically, execute the following MODIFY (or **F**) console command:

```
F PKISERVD,LOG sub-component.level[,sub-component.level…]
```

subcomponent.level
Sets the message level settings for the subcomponents. Use one of the subcomponents and message levels listed previously.

Displaying log options settings

To display the current logging options, execute the following MODIFY (or **F**) console command:

```
F PKISERVD,DISPLAY
```

The result of this command is information message IKYP025I. **Sample output:**

```
SY1  IKYP025I PKI SERVICES SETTINGS:
  CA DOMAIN NAME: Customers
  SUBCOMPONENT                  MESSAGE LEVEL
     POLICY                     WARNING MESSAGES AND HIGHER
     LDAP                       ERROR MESSAGES AND HIGHER
     SAF                        WARNING MESSAGES AND HIGHER
     DB                         INFORMATIONAL MESSAGES AND HIGHER
     CORE                       WARNING MESSAGES AND HIGHER
     PKID                       VERBOSE DIAGNOSTIC MESSAGES AND HIGHER
     TPOLICY                    WARNING MESSAGES AND HIGHER
  MESSAGE LOGGING SETTING: STDOUT_LOGGING
  CONFIGURATION FILE IN USE:
/etc/pkiserv/pkiserv.conf
  TEMPLATE FILE IN USE:
/etc/pkiserv/pkiserv.tmpl
  CA CERTIFICATE FINGERPRINTS:
   SHA1:   25:25:EE:1E:B1:3B:5D:92:E5:3B:74:0D:0C:F3:C3:93:F5:9A:78:ED
   SHA256: F6:C7:C4:B2:4F:99:1F:01:04:C2:AC:07:B3:8B:4A:69:
           78:6E:46:0C:95:E9:93:D9:BF:0A:F1:C1:49:C8:AD:2E
   SHA512: 84:A3:E8:2E:67:2E:F2:16:30:A8:ED:92:32:1D:F5:5F:
           14:58:63:B9:2B:B6:4F:B4:60:59:32:8C:69:8B:B7:CA:
           5C:AD:7D:C8:EF:6F:F9:20:5E:1B:B7:42:79:1C:53:1C:
           11:2D:5E:DD:25:A9:32:3A:3C:D2:CD:14:DB:B6:CA:65
  FIPS LEVEL: FIPS 140-2
  STATUS: OPERATIONAL
```

Note: The MD5 fingerprint is only displayed if FIPS mode is off.

The PKI administrator can use this command to display the fingerprints of the PKI Services CA certificate in support of Simple Certificate Enrollment Protocol (SCEP) certificate requests.

Part 8. Reference information

This part provides reference information, including listings of code samples for certain important files.

Note: The listings in this part might not be identical to the code samples shipped with the product. For the most current sample, see the appropriate source directory.

- Chapter 26, "Messages," on page 547 explains PKI Services messages.
- Chapter 27, "File directory structure," on page 603 describes product and file system directories for PKI Services and files contained in them.
- Chapter 29, "Environment variables," on page 615 explains the `pkiserv.envars` environment variables file and provides a code sample.
- Chapter 30, "The IKYSETUP REXX exec," on page 619 explains the contents of the IKYSETUP REXX exec that performs RACF administration and provides a code sample.
- Chapter 31, "Other code samples," on page 649 provides additional code samples. Table 103 on page 543 summarizes information about these code samples and those in the preceding chapters, summarizing their use, directory location, and the location where the code sample begins.
- Chapter 32, "SMF recording," on page 683 describes the content of the System Management Facility (SMF) record that is generated by PKI Services.

Table 103. Summary of information about important files

File	Description	Source location (default)	For code sample...
`httpd.conf`, `vhost80.conf`, `vhost443.conf`, and `vhost1443.conf`	Contains IBM HTTP Server - Powered by Apache directives.	`pki-install-dir/samples`	See "IBM HTTP Server - Powered by Apache configuration directives" on page 649.
IKYCDB2	Sample to create Db2 objects for the object store and issued certificate list (ICL). This JCL is used for `version 0` object store and ICL.	`SYS1.SAMPLIB`	See "IKYCDB2" on page 655.
IKYCDBV1	Sample to create Db2 objects for the object store and issued certificate list (ICL). This JCL is used for `version 1` object store and ICL.	`SYS1.SAMPLIB`	See "IKYCDBV1" on page 658.
IKYCVSAM	Sample IDCAMS JCL to create VSAM data sets (regardless of whether you are using a sysplex or non-sysplex). This JCL is used for `version 0` object store and ICL.	`SYS1.SAMPLIB`	See "IKYCVSAM" on page 661.
IKYCVSV1	Sample IDCAMS JCL to create VSAM data sets (regardless of whether you are using a sysplex or non-sysplex). This JCL is used for `version 1` object store and ICL.	`SYS1.SAMPLIB`	See "IKYCVSV1" on page 665.

Table 103. Summary of information about important files (continued)

File	Description	Source location (default)	For code sample...
IKYRVSAM	Sample IDCAMS JCL to add VSAM record-level sharing (RLS) support. IKYRVSAM reallocates your VSAM data sets in preparation for sharing in a sysplex. This JCL is used for `version 0` object store and ICL.	`SYS1.SAMPLIB`	See "IKYRVSAM" on page 669.
IKYRVSV1	Sample IDCAMS JCL to add VSAM record-level sharing (RLS) support. IKYRVSV1 reallocates your VSAM data sets in preparation for sharing in a sysplex. This JCL is used for `version 1` object store and ICL.	`SYS1.SAMPLIB`	See "IKYRVSV1" on page 673.
IKYSBIND	Sample job to create the Db2 package and plan for the object store and issued certificate list (ICL).	`SYS1.SAMPLIB`	See "IKYSBIND" on page 677.
IKYSETUP	REXX exec to set up RACF profiles.	`SYS1.SAMPLIB`	See Chapter 30, "The IKYSETUP REXX exec," on page 619.
IKYSGRNT	Sample job to grant EXECUTE authority for the Db2 package to the PKI Services daemon user ID.	`SYS1.SAMPLIB`	See "IKYSGRNT" on page 679.
IKYVBKUP	Sample JCL to back up the PKI Services VSAM data sets using the DFSMSdss DUMP utility.		See "IKYVBKUP" on page 680.
IKYVREST	Sample JCL to restore the PKI Services VSAM data sets from a backup taken with the DFSMSdss DUMP utility.		See "IKYVBKUP" on page 680.
`pkiserv.conf`	PKI Services configuration file.	`/usr/lpp/pkiserv/samples/` (You copy this file to the runtime directory, `/etc/pkiserv`.)	See Chapter 28, "The pkiserv.conf configuration file," on page 607.
PKISERVD	Sample procedure to start PKI Services daemon.	`SYS1.PROCLIB`	See "PKISERVD sample procedure to start PKI Services daemon" on page 682.
`pkiserv.envars`	PKI Services environment variables file.	`/usr/lpp/pkiserv/samples/` (You might need to copy this file to the runtime directory, `/etc/pkiserv`.)	See "The pkiserv.envars environment variables file" on page 617.
`pkiserv.tmpl`	PKI Services certificate template file for REXX CGI execs.	`/usr/lpp/pkiserv/samples/` (You copy this file to the runtime directory, `/etc/pkiserv`.)	Not provided.

Table 103. Summary of information about important files (continued)

File	Description	Source location (default)	For code sample...
`pkitmpl.xml`	PKI Services certificate template file for JavaServer pages.	`/usr/lpp/pkiserv/samples/` (You copy this file to the runtime directory, `/etc/pkiserv`.)	Not provided.
`pkixgen.tmpl`	PKI Services certificate template file for the daemon's use when you implement the web application using JavaServer pages.	You create this file from `pkitmpl.xml` using the TemplateTool utility.	Not provided.

Chapter 26. Messages

PKI Services message numbers begin with the three-character component prefix (**IKY**), followed by a fourth character that identifies the subcomponent. The following table lists the characters representing various subcomponents and describes where the messages appear.

Table 104. Meaning of fourth character in message number

Character	Meaning	Component producing messages	Where messages appear
C	CORE	Core subcomponent	PKI Services log
D	DB	Database accessing subcomponent	PKI Services log
I	INTERFACE	PKISERV CGIs or JSPs	In the user's web browser window
L	LDAP	LDAP bind subcomponent	PKI Services log
O	POLICY	Certificate creation and revocation policy subcomponent	PKI Services log
P	PKID	PKI Services daemon address space controller	• PKI Services log • (For those with destination and routing codes) operators console
S	SAF	SAF interfacing subcomponent	PKI Services log
U	UTILITY	Utility programs	UNIX standard error (`stderr`)

Characters five through seven are numeric. The eighth character is the message type:

Table 105. Meaning of eighth character in message number

Character	Meaning	Action required
I	Informational (status message)	No action required
E	Eventual action	Possible problem that might require eventual action
A	Action required	Problem that requires immediate attention

For information about setting messages options using environment variables, see "_PKISERV_MSG_LOGGING" on page 616.

IKYC001I **Error *nnnn action-being-performed*: *error-code-description***

Explanation

PKI Services is processing a request and has encountered an internal error. The action being performed and the error code encountered are displayed. A description of the error is also displayed, if known.

System action

The request is not processed.

User response

Report the error to the IBM support center.

IKYC002I **Error *nnnn* returned from CP_NewCertCreate: *error-code-description***

Explanation

PKI Services is attempting to create a certificate and has encountered an internal error. The action being performed and the error code encountered are displayed. A description of the error is also displayed, if known.

System action

The certificate is not created.

System programmer response

Report the error to the IBM support center.

IKYC003I **Error *nnnn* registering the next CRL cutting job: *error-code-description***

Explanation

PKI Services finished creating the current CRL and is attempting to schedule the next CRL creation thread. An error was encountered. The error code encountered is displayed. A description of the error is also displayed, if known.

System action

Future CRLs are not created until the problem is corrected and PKI Services is restarted.

System programmer response

Look for other error messages that can be issued such as IKYC011I. If no other messages were issued, report the error to the IBM support center.

IKYC004I **Error *nnnn* creating and sending CRLs: *error-code-description***

Explanation

PKI Services is attempting to create the current CRL and has encountered an error. The error code encountered is displayed. A description of the error is also displayed, if known. Note: If the error code is an LDAP return code, no error description is displayed. This would indicate a problem posting the CRL to the LDAP directory.

System action

If the CRL was created and the post to LDAP was unsuccessful, the post request remains in the PKI Services request database to be reattempted later. If posting continues to be unsuccessful for one week, the information is removed from the request database and deleted. For all other errors, PKI Services tries again to create the CRL during the next CRL interval.

System programmer response

If this is a problem with posting to LDAP, you should also see messages IKYC007I or IKYC008I or both. If so, follow the instructions for these messages. Otherwise, report the error to the IBM support center.

IKYC005I **Error *nnnn* posting {User | CA} Certificate to LDAP for distinguished-name: *error-code-description***

Explanation

PKI Services is attempting to post a certificate to the LDAP directory and has encountered an error. The distinguished name for which the post was attempted and the error code encountered are displayed. A description of the error is also displayed, if known. If the error code is an LDAP return code, no error description is displayed.

System action

If the post is unsuccessful for a given certificate, retries the post at the next post interval. If the post continues to be unsuccessful after 3 attempts, the post frequency for the certificate is reduced to no more than once per hour. After 26 unsuccessful attempts, it is further reduced to no more than once per day. After 33 unsuccessful attempts, the post request for the certificate is deleted from the request database.PKI Services

System programmer response

Determine whether the error occurred on the call to LDAP or within PKI Services, based on the presence of an error code description in the message. If no error code description is displayed in the message, the error occurred on the call to LDAP. If the error code is `LDAP_NO_SUCH_OBJECT`, the LDAP entry could not be created because the required suffix does not exist. Check the message to determine the entry that could not be created. If the entry should be posted to LDAP, you need to define the suffix in the LDAP server configuration file, and then stop and restart the LDAP server. For all other LDAP errors, follow the instructions in *z/OS IBM Tivoli Directory Server Client Programming for z/OS*. If an error code description is displayed in the message, the error occurred within PKI Services.

If the error code description is `Missing LDAP information`, then the `CreateOUValue` directive is missing from the LDAP section of the PKI Services configuration file. Add the directive, then stop and restart PKI Services. See Chapter 8, “Tailoring the PKI Services configuration file for LDAP,” on page 107 for more information.

Report any other PKI Services error to the IBM support center. If message IKYC009I is also displayed, report that information also.

IKYC007I **Error *nnnn* posting {CRL | ARL} to LDAP: *error-code-description***

Explanation

PKI Services is attempting to post a CRL or ARL to the LDAP directory and has encountered an error. The error code encountered is displayed. A description of the error is also displayed, if known. Note: If the error code is an LDAP return code, no error description is displayed.

System action

The post request remains in the PKI Services request database to be reattempted later. If posting continues to be unsuccessful for one week, the information is removed from the request database and deleted.

System programmer response

If the error is `LDAP_NO_SUCH_OBJECT`, the LDAP entry to contain the CRL or ARL does not yet exist. This is expected if you are starting PKI Services for the first time. For all other LDAP errors, follow the instructions in *z/OS IBM Tivoli Directory Server Client Programming for z/OS*. Report errors to the IBM support center. If message IKYC009I is also displayed, report that information also.

IKYC008I **Error *nnnn* creating an entry for {CA Certificate | User Cert | CRL | ARL} to LDAP for *distinguished-name*: *error-code-description***

Explanation

PKI Services is attempting to post a certificate, CRL, or ARL to the LDAP directory and has encountered an error. The distinguished name for which the post was attempted and the error code encountered are displayed. A description of the error is also displayed, if known. Note: If the error code is an LDAP return code, no error description is displayed.

System action

The post request remains in the PKI Services request database to be reattempted later. If posting continues to be unsuccessful for one week, the information is removed from the request database and deleted.

System programmer response

You can also see message IKYC005I or IKYC007I displayed. If so, follow the instructions for the message displayed. Follow related instructions in *z/OS IBM Tivoli Directory Server Client Programming for z/OS*. Report errors to the IBM support center. If message IKYC009I is also displayed, report that information also.

IKYC009I **LDAP post unsuccessful for object id = *nnnn*, state = *nnnn*, status = *nnnn*: *status-code-description***

Explanation

This message appears as supplemental information for messages IKYC005I and IKYC008I.

System programmer response

If reporting message IKYC005I or IKYC008I to the IBM support center, report this information also.

IKYC010I **Error *nnnn* returned from action-being-performed: *error-code-description***

Explanation

PKI Services is processing a request and has encountered an error. The action being performed and the error code encountered are displayed. A description of the error is also displayed, if known.

System action

The request is not processed.

System programmer response

Check the PKI Services logs for additional information about this error. You can use the _PKISERV_MSG_LEVEL environment variable to increase the messages that PKI Services generates. For more information about the logs and _PKISERV_MSG_LEVEL, see Chapter 25, “Using information from the PKI Services logs,” on page 537.

If the error code description is not self-explanatory, and you cannot find additional information in the logs, report the error to the IBM support center.

IKYC011I **Bad TimeBetweenCRLs value in pkiserv.conf file: *incorrect-value***

Explanation

PKI Services is reading its configuration file to locate the value that is specified for `TimeBetweenCRLs` in the **CertPolicy** section. The value that is specified has an incorrect syntax.

System action

CRL processing is suspended until the problem is corrected and PKI Services is restarted.

System programmer response

Correct the value and restart PKI Services. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC012I **Bad CRLDuration value in pkiserv.conf file: *incorrect-value***

Explanation

PKI Services is reading its configuration file to locate the value that is specified for `CRLDuration` in the **CertPolicy** section. The value that is specified has an incorrect syntax.

System action

CRL processing is suspended until the problem is corrected and PKI Services is restarted.

System programmer response

Correct the value and restart PKI Services. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC013I **Bad CreateInterval value in pkiserv.conf file**

Explanation

PKI Services is reading its configuration file to locate the value that is specified for `CreateInterval` in the **CertPolicy** section. The value that is specified has an incorrect syntax.

System action

PKI Services uses the default value of 3 minutes.

System programmer response

Correct the value and restart PKI Services if you want. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC014I **Bad RemoveCompletedReqs or RemoveInactiveReqs value in pkiserv.conf file**

Explanation

PKI Services is reading its configuration file to locate the value that is specified for either `RemoveCompletedReqs` or `RemoveInactiveReqs` in the **ObjectStore** section. The value that is specified has an incorrect syntax.

System action

Completed and inactive requests are not removed until the problem is corrected and PKI Services is restarted.

System programmer response

Correct the value and restart PKI Services. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC015I **Bad PostInterval value in pkiserv.conf file**

Explanation

PKI Services is reading its configuration file to locate the value that is specified for `PostInterval` in the **LDAP** section. The value that is specified has an incorrect syntax.

System action

PKI Services uses the default value of 5 minutes.

System programmer response

Correct the value and restart PKI Services if you want. For more information, see "Steps for tailoring the LDAP section of the configuration file" on page 108.

IKYC016I ***action-being-performed* returned *nnnn* in sub-function: *error-code-description***

Explanation

PKI Services is processing a request and has encountered an internal error. The action being performed, the subfunction that returned the error, and the error code encountered are displayed. A description of the error is also displayed, if known.

System action

The request is not processed.

System programmer response

Report the error to the IBM support center.

IKYC017I **JNH_inquire_certreq_startdate (*object-id*) found neither certificate request nor response (*nnnn*): *error-code-description***

Explanation

PKI Services is processing the start date in a request and has encountered an internal error. The request's ID and the error code encountered are displayed. A description of the error is also displayed, if known.

System action

The request is not processed.

System programmer response

Report the error to the IBM support center.

IKYC018I **{read | get_value} of *certificate-or-CRL-extension-name* returned *nnnn*: *error-code-description***

Explanation

PKI Services is processing a CRL or certificate extension field and has encountered an internal error. The field name and the error code encountered are displayed. A description of the error is also displayed, if known.

System action

The CRL or certificate is not processed.

System programmer response

Report the error to the IBM support center.

IKYC020I **Retrieving CA value failed *nnnn*: *error-code-description***

Explanation

PKI Services is processing a certificate extension field in preparation of posting the certificate to the LDAP directory. The processing has encountered an internal error. The error code encountered is displayed. A description of the error is also displayed, if known.

System action

The certificate is not posted to the LDAP directory.

System programmer response

Report the error to the IBM support center.

IKYC021I **CRL claims to have only User and only CA certs**

Explanation

PKI Services is processing a CRL extension field in preparation of posting the CRL to the LDAP directory. The processing has encountered an internal error. The error code encountered is displayed. A description of the error is also displayed, if known.

System action

The CRL is not posted to the LDAP directory.

System programmer response

Report the error to the IBM support center.

IKYC022I **Invalid type for object *object-id* in JNH_set_revreq_invalidityDate: *error-code-description***

Explanation

PKI Services is processing a revocation request and has encountered an internal error. The revocation request's ID and the error code encountered are displayed. A description of the error is also displayed, if known.

System action

The revocation request is not processed.

System programmer response

Report the error to the IBM support center.

IKYC023I **Request index (*index-number*) greater than number of revocations (*nnnn*) in JNH_set_revreq_invalidityDate**

Explanation

PKI Services is processing a revocation request and has encountered an internal error.

System action

The revocation request is not processed.

System programmer response

Report the error to the IBM support center.

IKYC024I **Failed to schedule event in *nnnn* seconds, status = *nnnn*: *error-code-description***

Explanation

PKI Services is attempting to schedule a timed event and has encountered an internal error. The error code encountered is displayed. A description of the error is also displayed, if known.

System action

The event is not scheduled.

System programmer response

Report the error to the IBM support center.

IKYC025I **Failed to schedule event status = *nnnn*: *error-code-description***

Explanation

PKI Services is attempting to schedule a timed event and has encountered an internal error. The error code encountered is displayed. A description of the error is also displayed, if known.

System action

The event is not scheduled.

System programmer response

Report the error to the IBM support center.

IKYC026I **Deleting {inactive | completed} object *object-id*. Last changed at *YYYY/MM/DD HH:MM:SS***

Explanation

PKI Services is attempting to purge the request database of inactive and completed requests. A request that has met the criteria for deletion has been found. The request's ID is displayed along with information about when it was last changed. This is an informational message only.

System action

The request is deleted. PKI Services continues normal processing.

IKYC027I **Removing certificate post request after *nnnn* unsuccessful attempts**

Explanation

PKI Services is attempting to purge the request database of unsuccessful LDAP post requests. A request that has met the criteria for deletion has been found. The number of unsuccessful attempts for this request is displayed. This is an informational message only.

System action

The request is deleted. PKI Services continues normal processing.

IKYC028I **Export for CertId *certificate-id* unsuccessful. Request is still pending approval or yet to be issued**

Explanation

A client has requested a certificate and is attempting to retrieve it. The retrieval was unsuccessful because the certificate is not yet available. The request either has yet to be approved by a PKI Services administrator or has been approved, but has not yet been issued by PKI Services. This is an informational message only.

System action

The state of the request is unchanged. PKI Services continues normal processing.

PKI Services administrator response

Use PKI Services administrative functions to query the request to check its state. If the request is still pending approval, determine whether the request should be approved or rejected and take action accordingly. For more information, see "Processing certificate requests" on page 409.

IKYC029I **Error: certificate request type is invalid for certificate creation**

Explanation

PKI Services is processing a certificate request and has encountered an internal error.

System action

The certificate request is not processed.

System programmer response

Report the error to the IBM support center.

IKYC030I **Error *nnnn* retrieving LDAP *attribute-name* attribute data from *distinguished-name*: *error-code-description***

Explanation

PKI Services is trying to retrieve some attribute data from an entry in the LDAP directory and has encountered an error. The attribute name and distinguished name for which the retrieve was attempted and the error code encountered are displayed. If known, a description of the error is also displayed.

Note: If the error code is an LDAP return code, no error description is displayed.

System action

If the attribute being retrieved is `'MAIL'`, PKI Services is trying to retrieve the client's email address to send the client a certificate expiration warning message or a renewed certificate. The warning message or the renewed certificate is not sent now but sending is tried later.

System programmer response

Follow related instructions in *z/OS IBM Tivoli Directory Server Client Programming for z/OS*. Report errors to the IBM support center.

IKYC031I **Error *nnnn* invoking sendmail with email address *email-address* retrieved from LDAP entry *distinguished-name***

Explanation

PKI Services tried to call the sendmail utility to notify a client that the certificate is expiring or is renewed. The call was unsuccessful. This message displays the email address and distinguished name from which it was retrieved and the error code encountered.

System action

It can be unclear if the warning message or the renewed certificate has been sent. If the email address appears to be genuine, PKI Services retries sending later.

System programmer response

Diagnose the problem by consulting *z/OS Communications Server: IP Diagnosis Guide* and related documents. Report non-Communications Server errors to the IBM support center.

IKYC032I **Error *nnnn* invoking sendmail with email address *email-address* provided by *distinguished-name***

Explanation

PKI Services is trying to notify a client that the certificate is ready, rejected, or recovered. Notification is accomplished by calling the sendmail utility. The call was unsuccessful. This message displays the email address and the subject's distinguished name from the request. The error code encountered is also displayed.

System action

It is unclear if the message has been sent.

System programmer response

Diagnose the problem by consulting *z/OS Communications Server: IP Diagnosis Guide* and related documents. Report non-Communications Server errors to the IBM support center.

IKYC033I **Error *nnnn* accessing {ReadyMessageForm | RejectMessageForm | ExpiringMessageForm | AdminNotifyForm | RenewCertForm} *form-value***

Explanation

PKI Services is attempting to notify a client that the certificate is ready, has been rejected, is expiring, is pending for approval, or has been renewed. The message to be sent is derived by reading the message form from a file or data set specified in the **General** section of the PKI Services configuration file. Either the file name was not specified correctly, or the file read was unsuccessful. The configuration file keyword in error is displayed. The name of the failing file or data set and the error code encountered are also displayed, if known. For ExpiringMessageForm, an error code of zero with no file or data set name displayed indicates that the keyword is required but is missing from the PKI Services configuration file.

System action

The message is not sent. If this is the expiring warning message, notify pending message or the renewed certificate message, it is attempted later.

System programmer response

Locate the failing `form-typeMessageForm` value in the `pkiserv.conf` file. Make sure that the value specifies the correct file or data set name and that the file or data set exists. If no errors are found, contact your RACF administrator to ensure that the user ID assigned to the PKI Services daemon has permission to open the file or data set for reading. After making a correction, restart PKI Services. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC034I **Error issuing DEQ for resource *resource-name*, return code was *return-code***

Explanation

PKI Services background certificate processing encountered an internal error trying to release control of a resource using the DEQ service. The resource name and return code from the DEQ macro are displayed.

System action

PKI Services processing continues. However, further processing of certificate requests can fail until PKI Services is stopped and restarted.

System programmer response

Stop and restart PKI Services. If the problem occurs again, report the error to the IBM support center.

IKYC035I Bad ExpireWarningTime value in pkiserv.conf file

Explanation

PKI Services is reading its configuration file to locate the value that is specified for `ExpireWarningTime` in the **CertPolicy** section. The value that is specified has an incorrect syntax.

System action

PKI Services continues, but no expiration warning messages are issued.

System programmer response

Correct the value and restart PKI Services if you want. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC036I Bad MaxSuspendDuration value in pkiserv.conf file

Explanation

PKI Services is reading its configuration file to locate the value that is specified for `MaxSuspendDuration` in the **CertPolicy** section. The value that is specified has an incorrect syntax.

System action

CRL processing continues. PKI Services processes as if the suspension grace period is unlimited.

System programmer response

Correct the value and restart PKI Services if you want. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC037I Bad RemoveExpiredCerts or RemoveExpiredCertsAndKeys value in pkiserv.conf file

Explanation

PKI Services is reading its configuration file to locate the value that is specified for `RemoveExpiredCerts` or `RemoveExpiredCertsAndKeys` in the **ObjectStore** section. The value that is specified has an incorrect syntax.

System action

PKI Services continues, but expired certificates are not removed from the issued certificate list (ICL).

System programmer response

Correct the value and restart PKI Services if you want. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC038I Deleting expired certificate with serial number *certificate-serial-number*

Explanation

PKI Services is attempting to purge the issued certificate list (ICL) of expired certificates. A certificate that has met the criteria for deletion has been found. The certificate's serial number is displayed. This is an informational message only.

System action

The request is deleted. PKI Services continues normal processing.

IKYC039I Bad CRLDistName value in pkiserv.conf file

Explanation

PKI Services is initializing and is reading its configuration file to locate the value that is specified for `CRLDistName` in the **CertPolicy** section. The value that is specified does not contain all alphanumeric characters.

System action

PKI Services stops.

System programmer response

Correct the value and restart PKI Services if you want. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC040I Bad CRLDistURI*n* value in pkiserv.conf file: LdapServer*n*

Explanation

PKI Services is reading its configuration file to locate the LDAP server that is specified in the **LDAP** section of the PKI Services configuration file. The value is to be used to create the URI format for the CRLDistributionPoints extension for the LDAP protocol. The `Server` value that is specified by LdapServer*n* cannot be found or contains incorrect information.

System action

PKI Services continues, but the URI format for that protocol distribution point is not created.

System programmer response

Ensure that the CRLDistURI*n* value locates the correct LdapServer*n* or BindProfile*n* value in the **LDAP** section or the default FACILITY class profile, IRR.PROXY.DEFAULTS. For profile values, ensure that the profile exists and contains the correct information. Correct the value and restart PKI Services if you want. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC041I Bad CRLDistURI*n* value in pkiserv.conf file, exceeds the number of LDAP servers: LdapServer*n*

Explanation

PKI Services is reading its configuration file to locate the LDAP server that is specified in the **LDAP** section of the PKI Services configuration file. The value is to be used to create the URI format for the CRLDistributionPoints extension for the LDAP protocol. The value *n* indicated in LdapServer*n* is greater than that specified by NumServers.

System action

PKI Services continues, but the URI format for that protocol distribution point is not created.

System programmer response

Correct the value and restart PKI Services if you want. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC042I Bad CRLDistURI*n* format in pkiserv.conf file: CRLDistURI*n*

Explanation

PKI Services is reading its configuration file to create the URI format for the CRLDistributionPoints extension. The value that is specified has incorrect syntax.

System action

PKI Services continues, but the URI format for that protocol distribution point is not created.

System programmer response

Correct the value and restart PKI Services if you want. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC043I Error *nnnn* in creating HFS file *file name* to store distribution point CRL

Explanation

PKI Services is trying to store the distribution point CRL in a file system file. An I/O error occurred during the processing.

System action

PKI Services continues, but the distribution point CRL is not created.

System programmer response

Fix the I/O error and wait for the next distribution point CRL to be created.

IKYC044I Bad OCSPType value in pkiserv.conf file: *value*

Explanation

PKI Services responder is reading its configuration file to check whether the OCSP responder is enabled when receiving an OCSP request. The expected value is either `'none'` or `'basic'`. The value that is specified is not one of these.

System action

The responder is not enabled. The client gets a response back with status `'Try later'`.

System programmer response

Correct the value and restart PKI Services if you want. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC045I Unknown section or keyword in pkiserv.conf file: Section: [*section*], Keyword *keyword*

Explanation

PKI Services is reading its configuration file during initialization. One of the following conditions occurred:

- An unknown section name was found.
- An unknown keyword was found.
- A valid keyword was placed in the wrong section.
- A keyword was placed before any sections were defined.

System action

The keyword is ignored and PKI Services continues.

System programmer response

Correct the section name or the keyword and restart PKI Services if you want. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC046I Incorrect encoding of SCEP {request | PKCS10}

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request from a SCEP client. The request for a PKI operation contains incorrect ASN.1 encoding.

System action

PKI Services rejects the SCEP request.

System programmer response

Correct the SCEP client to produce a correctly encoded ASN.1 request. Report the error to the support center for the provider of your SCEP client.

IKYC047I Incorrect signature on SCEP {request | PKCS10}

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request from a SCEP client. The request for a PKI operation or its enclosed PKCS #10 certificate request is incorrectly signed or was altered.

System action

PKI Services rejects the SCEP request.

System programmer response

Correct the SCEP client to produce a correctly signed request. Report the error to the support center for the provider of your SCEP client.

IKYC048I Unsupported algorithm in SCEP {request | PKCS10 | signing certificate}

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request from a SCEP client. The request, the PKCS #10 certificate request enclosed within it, or the signing certificate enclosed within it contains an algorithm identifier or a key size that is not supported by PKI Services at the FIPS level in which PKI Services is operating.

System action

PKI Services rejects the SCEP request.

System programmer response

Correct the SCEP client to produce algorithms and key sizes that are supported at the FIPS level that is used by PKI Services. Report the error to the support center for the provider of your SCEP client. For more information, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333.

IKYC049I SCEP signing certificate is {expired | revoked | not known to PKI Services}

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request from a SCEP client. The request for a PKI operation contains a signing certificate that is expired, revoked, or unknown to PKI Services. The signing certificate cannot be used to authenticate the SCEP client.

System action

PKI Services rejects the SCEP request.

System programmer response

Determine whether the SCEP client should request certificates from PKI Services. If so, correct the SCEP client to use a valid signing certificate previously issued by PKI Services. If none exists, reconfigure the SCEP client to remove any existing certificates and start the certificate request from the beginning using a new key-pair and a new self-signed certificate. Report the error to the support center for the provider of your

SCEP client. For more information, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333.

IKYC050I **Requested SCEP certificate is {expired | revoked}**

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request from a SCEP client. The request for a PKI operation requests a certificate that was previously issued by PKI Services but is revoked or expired, as indicated by the message displayed. The requested certificate cannot be used by the SCEP client.

System action

PKI Services rejects the SCEP request.

System programmer response

Determine whether the SCEP client should request certificates from PKI Services. If so, reconfigure the SCEP client to remove any existing certificate request entries and start the certificate request from the beginning using a new key-pair and a new self-signed certificate. Report the error to the support center for the provider of your SCEP client. For more information, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333.

IKYC051I **Error 0x*nnnn* decrypting SCEP request**

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request for a PKI operation from a SCEP client. PKI Services is attempting to recover the encrypted portion of the request using System SSL services but is unable to do so. The error code that is returned by System SSL is displayed in the message.

System action

PKI Services rejects the SCEP request.

System programmer response

Look up the **0x*nnnn*** error code in *z/OS Cryptographic Services System SSL Programming*. Most likely, the SCEP client incorrectly encrypted the request, for example by encrypting it for a different host server. Report the error to the support center for the provider of your SCEP client. For more information, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333.

IKYC052I **Unsupported SCEP message type: *nn***

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request for a PKI operation from a SCEP client. The message type that is displayed in the message is either unrecognized or unsupported by PKI Services. PKI Services supports only message types PKCSReq (19) and `GetCertInitial` (20).

System action

PKI Services rejects the SCEP request.

System programmer response

Reconfigure the SCEP client to produce message types that are supported by PKI Services. Report the error to the support center for the provider of your SCEP client. For more information, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333.

IKYC053I **SCEP key or name mismatch. SCEP transaction ID: *SCEP-transaction-ID***

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request for a PKI operation from a SCEP client. While processing the displayed SCEP request transaction ID, PKI Services found one of the following inconsistencies:

- The signing certificate is self-signed but the public key within it does not match the key in the enclosed PKCS #10 request.
- The signing certificate's subject name does not match the subject name in the PKCS #10 request or the subject portion of the `IssuerAndSubject` field in the SCEP request.
- The signing certificate is not self-signed and not issued by PKI Services.
- The issuer portion of the `IssuerAndSubject` field in the SCEP request does not identify PKI Services.

System action

PKI Services rejects the SCEP request.

System programmer response

Correct the SCEP client to remove the inconsistency. Report the error to the support center for the provider of your SCEP client. For more information, see Chapter

15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333.

IKYC054I **Incorrect reuse of SCEP transaction ID by client name: *client-name***

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request for a PKI operation from a SCEP client that is named in the message. The request contains a SCEP transaction ID that was previously used in a different request. Transaction IDs must uniquely identify a transaction and cannot be reused.

System action

PKI Services rejects the SCEP request.

System programmer response

Determine whether the SCEP client should request certificates from PKI Services. If so, reconfigure the SCEP client to remove any existing certificate request entries and start the certificate request from the beginning using a new key-pair and a new self-signed certificate. Report the error to the support center for the provider of your SCEP client. For more information, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333.

IKYC055I **Bad PREREGISTER section in certificate template, nickname: *template-nickname***

Explanation

PKI Services detects a problem in a template PREREGISTER section in one of the following circumstances:

1. When PKI Services is starting, it finds the PRERESISTER section for the Simple Certificate Enrollment Protocol (SCEP) or Enrollment over Secure Transport (EST) template is incorrectly terminated.
2. When PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request or an EST request for a PKI operation from a SCEP or EST client, the named certificate template that is used to preregister the SCEP client contains one of the following errors that are related to its PREREGISTER section:
   - The PREREGISTER section does not exist.
   - The PREREGISTER section is incorrectly terminated.
   - A required directive is missing.
   - A directive has an incorrect value.

System action

PKI Services rejects the SCEP or EST request.

System programmer response

Edit the PKI Services certificate templates file (`pkiserv.tmpl`) and correct the error. For more information, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333 or Chapter 16, "Using Enrollment over Secure Transport (EST)," on page 341.

IKYC056I **Template missing from certificate templates file, nickname: *template-nickname***

Explanation

PKI Services cannot find the template nickname in the templates file in one of the following circumstances:

1. PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) or PKI Services is processing an Enrollment over Secure Transport (EST) request for a PKI operation from an EST client, and the named certificate template that is used to preregister the SCEP or EST client no longer exists or the PREREGISTER section is incorrectly terminated.
2. During the automatic renewal process, the named certificate template that is used to generate the original certificate no longer exists.
3. PKI Services is trying to construct a quick link which contains the transaction ID and the certificate template name in the email message (Ready message) to notify a requester that the certificate is ready to be picked up. The certificate template that is identified by the nickname indicated in the message cannot be found. If the indicated nickname is <NONICK>, the <NICKNAME=>directive was missing at the time the request was submitted; otherwise, it had been there at the time of submission but was removed later.

System action

For case "1" on page 558, PKI Services rejects the SCEP or EST request. For case "2" on page 558, the certificate cannot be automatically renewed. For case "3" on page 558, the constructed quick link in the Ready message does not work and the certificate requester needs to use the other link to pick up the certificate, which requires manual input of certificate type and transaction ID.

System programmer response

For case "1" on page 558, edit the PKI Services certificate templates file (`pkiserv.tmpl`) and add the template. For more information, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333 or Chapter 16, "Using Enrollment over Secure Transport (EST)," on page 341. For case "2" on page 558, restore the nickname in the corresponding template and wait for the next automatic renewal processing. For case "3" on page 558, make sure that the template section has the <NICKNAME=> directive.

IKYC057I **SCEP request for client name *client-name* rejected by {SemiauthenticatedClient | UnauthenticatedClient | SubsequentRequest | RenewalRequest} directive.**

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request for a PKI operation from a SCEP client. The certificate template that is used to preregister the named SCEP client contains a directive in the PREREGISTER section indicating that the request should be rejected. The directive is displayed in the message.

System action

PKI Services rejects the SCEP request. The rejected certificate request is recorded in the request database.

System programmer response

Determine whether the SCEP client should request certificates from PKI Services. If so, reconfigure the SCEP client to remove any existing certificate request entries and start the certificate request from the beginning using a new key-pair and a new self-signed certificate. Create or re-create a PKI Services preregistration record as needed. Also, ensure that the SCEP client is using a subject name that is consistent with the preregistration record. Make corrections to the SCEP client or delete and re-create the preregistration record as needed. For more information, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333.

IKYC058I **No preregistration record found for {SCEP | EST} request, client name: *client-name***

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request or an Enrollment over Secure Transport (EST) request for a PKI operation from the named SCEP or EST client. No preregistration record matching the SCEP or EST client was found.

System action

PKI Services rejects the SCEP or EST request.

System programmer response

Determine whether the SCEP or EST client should request certificates from PKI Services. If so, ensure that a preregistration record exists for the SCEP or EST client and that the client is using a client name that matches the preregistration record. Make corrections to the SCEP or EST client, or delete and re-create the preregistration record as needed. For more information, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333 or Chapter 16, "Using Enrollment over Secure Transport (EST)," on page 341.

IKYC059I **No previous SCEP request found for SCEP GetCertInitial, client name: *client-name***

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request from the named SCEP client. The request is `GetCertInitial` (a polling operation to pick up a previously requested certificate). PKI Services is unable to locate the previous SCEP request.

System action

PKI Services rejects the SCEP request.

System programmer response

Determine whether the SCEP client should request certificates from PKI Services. If so, reconfigure the SCEP client to remove any existing certificate request entries and start the certificate request from the beginning using a new a key-pair and new self-signed certificate. Make corrections to the SCEP client, or delete and re-create the preregistration record as needed. For more information, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333.

IKYC060I **CONSTANT section of certificate template, nickname: *template-***

nickname* contains an incorrect value: *field-name=value

Explanation

PKI Services is processing a Simple Certificate Enrollment Protocol (SCEP) request for a PKI operation from a SCEP client. The certificate template that is used to preregister the SCEP client is in error. A field found in the **<CONSTANT>** section of the template contains a missing or incorrect value. The nickname of the certificate template and the incorrect value are displayed in the message.

System action

PKI Services rejects the SCEP request.

System programmer response

Edit the certificate templates file and correct the error. For more information, see Chapter 15, “Enabling Simple Certificate Enrollment Protocol (SCEP),” on page 333.

IKYC061I cannot generate a certificate or a request with automatic renewal

Explanation

PKI Services is trying to process a GENCERT/REQCERT or GENRENEW/REQRENEW request. The <AUTORENEW=Y or N>tag is specified at the correct position in its template, but NotifyEmail has no input.

System action

PKI Services does not process the request.

System programmer response

Enforce the input of NotifyEmail.

IKYC062I cannot enable or set up automatic renewal service for this request or certificate

Explanation

PKI Services is trying to enable or set up automatic certificate renewal for this request or certificate. But NotifyEmail is not in the request, or in the case of a certificate, one of the following has occurred:

- The certificate was created without NotifyEmail specified.
- The certificate has already been renewed.
- The expiration warning notification has already been sent for this certificate.

System action

PKI Services does not process the request.

System programmer response

None

IKYC063I The list of the pending approval requests may not be complete

Explanation

PKI Services is trying to construct a list of requests that are pending administrator approval. An error occurred when attempting to add an entry to the list.

System action

The request is not added to the pending request list.

System programmer response

None.

IKYC064I Error *nnnn* invoking sendmail to notify the administrator with email address *email-address* of requests awaiting approval

Explanation

PKI Services is trying to notify an administrator that one or more certificate requests are waiting for approval, and called the sendmail utility, which failed with the return code indicated in the error message. This error code can indicate an improperly defined email address, a sendmail configuration error, or a network interruption.

System action

It is unclear if the message has been sent.

System programmer response

Diagnose the problem by consulting *z/OS Communications Server: IP Diagnosis Guide* and related documents. Report non-Communications Server errors to the IBM support center.

IKYC065I The value value that is specified for tag *tag* in template *template-nickname* is ignored

Explanation

PKI Services is processing a tag and its value in the templates file, and either the tag is not in the expected position or it does not contain an expected value. The message indicates problem tag, its value, and the template nickname.

System action

PKI Services ignores the tag.

System programmer response

None.

IKYC066I **cannot find the template for GENRENEW or REQRENEW with nickname: *template-nickname***

Explanation

PKI Services is processing a GENRENEW or REQRENEW request and cannot find the template with the indicated nickname.

System action

PKI Services renews a certificate with no automatic renewal set up.

System programmer response

None

IKYC067I **An attempt to check whether the templates file has been updated failed**

Explanation

PKI Services attempted to determine whether the templates file has been updated since it was last read, but encountered an error.

System action

PKI Services opens and reads the templates file again to ensure that it uses the current values in the file.

System programmer response

None.

IKYC068I **The templates file used may not be current**

Explanation

If the web application was implemented using REXX CGI execs, PKI Services determined that it needed to read the templates file, but was unable to either open or read the file.

If the web application was implemented using JavaServer Pages (JSPs), the XML template file `pkitmpl.xml` has been updated, but the file `pkixgen.tmpl` has not been regenerated.

System action

PKI Services uses the last updated version of the templates file.

System programmer response

If the web application was implemented using JavaServer Pages (JSPs), use the TemplateTool utility to create an updated copy of `pkixgen.tmpl`.

IKYC069I **The renewed certificate with transaction ID *transaction-id* cannot be sent to email address *email-address***

Explanation

PKI Services attempted to create or retrieve the note with the renewed certificate. An internal error occurred.

System action

None

System programmer response

Record the information from the message and manually send the transaction ID to the email address to tell the user to pick up the renewed certificate.

IKYC070I **The exit program {*exit-program*} was canceled for {*pre / post*} processing**

Explanation

PKI Services is executing the specified exit program. The program did not return within the time limit that is specified by the ExitTimeout keyword in the configuration file for the PKI Services daemon (`pkiserv.conf`).

System action

PKI Services postpones the automatic renewal processing until the next day.

System programmer response

Check the exit program to see whether there are any problems. If you determine that it needs a longer time to run, adjust the ExitTimeout value in the `pkiserv.conf` file.

IKYC071I **Bad ExitTimeout value in pkiserv.conf file: {*value*}**

Explanation

PKI Services is reading its configuration file to locate the value that is specified for ExitTimeout in the General section. The value that is specified is incorrect.

System action

PKI Services continues. If the value that is specified is greater than 1 hour, then PKI Services uses 1 hour for the ExitTimeout value when the automatic renewal exit is invoked, otherwise it uses the default value of 30 seconds.

System programmer response

Correct the ExitTimeout value and restart PKI Services. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC072I **Unexpected return value from automatic renewal exit program {*exit-program*}: {*unexpected-value*}**

Explanation

PKI Services invoked the automatic certificate renewal preprocessing exit program specified, which returned an unexpected return value.

System action

PKI Services treats the unexpected return value as if a return value of 4 was returned. The automatic renewal of the certificate is deferred until the next time automatic renewal is performed.

System programmer response

Update the exit program to ensure that it returns only the documented expected values.

IKYC073I **A problem was encountered during automatic renewal {*pre* | *post*} exit processing in program {*exit program*}**

Explanation

PKI Services has encountered an error during automatic renewal exit processing. For example, it cannot invoke the specified program or it cannot retrieve the status from the exit program.

System action

PKI Services postpones the automatic renewal processing until the next day.

System programmer response

If diagnostic level messages are enabled for the CORE component, additional diagnostic level messages are written to the log to help identifying the exact cause of this failure. If diagnostic level messages were not enabled at the time of the error, enable diagnostic level messages for the CORE component and re-create the error. For information about enabling diagnostic level messages, see "_PKISERV_MSG_LEVEL subcomponents and message levels" on page 541.

IKYC074I **Error *nnnn* accessing the required {ReadyMessageForm | RecoverForm} *form-value***

Explanation

PKI Services is attempting to process a request or a certificate that involves a key that is generated by PKI Services. The processing requires the setup of email notification to notify a client of the transaction ID or the key ID so that the client can pick up the certificate through a provided link. The message to be sent is derived by reading the message form from a file or data set specified in the **General** section of the PKI Services configuration file. Either the file name was not specified correctly, or the file read was unsuccessful. The configuration file keyword in error is displayed. The name of the failing file or data set and the error code encountered are also displayed, if known.

System action

The request is not processed.

System programmer response

Locate the failing ReadyMessageForm or RecoverForm value in the `pkiserv.conf` file. Make sure that the value specifies the correct file or data set name and that the file or data set exists. If no errors are found, contact your RACF administrator to ensure that the user ID assigned to the PKI Services daemon has permission to open the file or data set for read. After making a correction, restart PKI Services. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC075I **Bad MaintRunTime value in pkiserv.conf file: {*value*}**

Explanation

PKI Services is reading its configuration file, `pkiserv.conf`, to locate the value that is specified for `MaintRunTime` in the **General** section. The value that is specified is incorrect.

System action

PKI Services continues processing, using the default value of midnight, `00:00`.

System programmer response

Correct the value of `MaintRunTime` and restart PKI Services. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC076I Bad MaintRunDays value in pkiserv.conf file: {*value*}

Explanation

PKI Services is reading its configuration file, `pkiserv.conf`, to locate the value that is specified for `MaintRunDays` in the **General** section. The value that is specified is incorrect.

System action

PKI Services continues processing, using the default value of every day of the week, `0123456`.

System programmer response

Correct the value of `MaintRunDays` and restart PKI Services. For more information about the `MaintRunDays` parameter, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC077I Incorrect DBType value in pkiserv.conf file: *value*

Explanation

PKI Services is reading its configuration file, `pkiserv.conf`, to locate the value that is specified for `DBType` in the **ObjectStore** section. The value that is specified is incorrect. The value of `DBType` must be `VSAM` or `DB2`.

System action

PKI Services stops.

System programmer response

Correct the value of `DBType` and restart PKI Services. For more information about the `DBType` parameter, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC078I Incorrect DBSubsystem value in pkiserv.conf file: *value*

Explanation

PKI Services is reading its configuration file, `pkiserv.conf`, to locate the value that is specified for `DBSubsystem` in the **ObjectStore** section. The value that is specified is incorrect. The value of `DBSubsystem` must be a 4-character Db2 subsystem name.

System action

PKI Services stops.

System programmer response

Correct the value of `DBSubsystem` and restart PKI Services. For more information about the `DBSubsystem` parameter, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC079I Incorrect DBPackage value in pkiserv.conf file: *value*

Explanation

PKI Services is reading its configuration file, `pkiserv.conf`, to locate the value that is specified for `DBPackage` in the **ObjectStore** section. The value that is specified is incorrect. The value of `DBPackage` must be the name of a Db2 package.

System action

PKI Services stops.

System programmer response

Correct the value of `DBPackage` and restart PKI Services. For more information about the `DBPackage` parameter, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC080I Error *nnnn* in creating UNIX file *file_name* to post CRL

Explanation

PKI Services is trying to store a CRL in the z/OS UNIX file system file *file_name* for posting. An I/O error occurred during the processing.

System action

PKI Services continues, but the file is not created. The CRL is not posted.

Fix the I/O error and wait for the next CRL to be created.

IKYC081I Error *nnnn* in reading UNIX file *file_name* to post CRL

Explanation

PKI Services is trying to read a CRL from the file that is named *file_name* in the z/OS UNIX file system. An I/O error occurred during the processing.

System action

PKI Services continues, but does not post the CRL.

System programmer response

Fix the I/O error and wait for the next posting interval. For information about the posting interval, see the description of the `PostInterval` configuration parameter in Table 25 on page 108.

IKYC082I Bad URI CRL posting path and DP name value in pkiserv.conf file.

Explanation

PKI Services is reading its configuration file to locate the values specified for `CRLDistDirPath` and `CRLDistName` to save CRLs in the z/OS UNIX file system. The combination of the values specified is not valid because it results in a path length greater than the maximum length of 240. The path length includes a trailing "/" character on `CRLDistDirPath` if it does not include one.

System action

PKI Services continues, but CRLs are not saved in the file system.

System programmer response

Correct the values of `CRLDistDirPath` and `CRLDistName` in the `pkiserv.conf` file, and ensure that the resulting path length (including a trailing "/" character on `CRLDistDirPath` if it does not include one) is not greater than 240. Then restart PKI Services.

IKYC083I Bad Large CRL posting path and DP name value in pkiserv.conf file.

Explanation

PKI Services is reading its configuration file to locate the values specified for `LargeCRLPostPath` and `CRLDistName` to save CRLs in the z/OS UNIX file system for LDAP posting. The combination of values specified is not valid because it results in a path length greater than the maximum length of 240. The path length includes a trailing "/" character on `LargeCRLPostPath` if it does not include one.

System action

PKI Services continues, but large CRL posting cannot be enabled. CRLs whose size is 32 KB or smaller are stored in the object store for posting. CRLs larger than 32 KB are not posted. (32 KB is an approximate value.)

System programmer response

Correct the values of `LargeCRLPostPath` and `CRLDistName` in the `pkiserv.conf` file. Ensure that the resulting path length (including a trailing "/" character on `LargeCRLPostPath` if it does not include one) is not greater than 240. Then, restart PKI Services.

IKYC084I Large CRL posting object found while large CRL posting is not enabled.

Explanation

During LDAP posting, PKI Services has found a CRL posting object larger than 32 KB, and `EnableLargeCRLPosting` has not been set to T in the `pkiserv.conf` file to enable posting of large CRLs. (32 KB is an approximate value.)

System action

PKI Services deletes the large CRL posting object from the object store and continues.

System programmer response

None.

IKYC086I Requests for CA certificates are prohibited by path length constraint.

Explanation:
The CA certificate in use has a path length constraint value of zero, which prohibits the creation of subordinate or intermediate CA certificates when the `EnablePathLenConstraint` keyword is set to T in the `pkiserv.conf` file. Because the certificate request includes certificate authority key usage bits (`keyCertSign` or `cRLSign`, or both), it is considered to be a request for a CA certificate.

System action:
The certificate request fails.

System programmer response
If this configuration was intended, restrict the requester from requesting the `keyCertSign` key usage. For example, remove the **keyusage** list from the PKI Services web page. If this configuration was not intended, perform one of the following actions and restart PKI Services:

- Disable path length constraint by removing the `EnablePathLenConstraint` keyword or setting its value to F.
- Reconfigure PKI Services to use a CA certificate that does not constrain the path length, or that has the path length greater than zero.

IKYC087I **Bad CRLWTONotification value in pkiserv.conf file: *value***

Explanation:
PKI Services is reading its configuration file, `pkiserv.conf`, to locate the value that is specified for `CRLWTONotification` in the **CertPolicy** section, to determine when to issue a console message when the CRL is available. The expected value is either none or `file`. The value that is specified is not one of these values.

System action:
No console message is issued to indicate the availability of the CRL.

System programmer response:
Correct the value of `CRLWTONotification` and restart PKI Services. For more information about the `CRLWTONotification` keyword, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC088I **CRL notification cannot be enabled**

Explanation
The `CRLWTONotification` variable is specified in the `pkiserv.conf` file. But neither of the following actions have been taken:

- Setting up CRL distribution points using the HTTP protocol
- Enabling large CRL processing

Therefore, CRL notification cannot be enabled.

System action:
No console message ID issued to indicate the availability of the CRL.

System programmer response:
Specify the HTTP protocol for `CRLDistURI1` or specify T for `EnableLargeCRLPosting` and restart PKI Services. For more information about these keywords, see "(Optional) Steps for updating the configuration file" on page 74. For information about setting up CRL distribution points, see "Customizing distribution point CRLs" on page 290. For information about enabling large CRL processing, see "Enabling support for large CRLs" on page 297.

IKYC089I **Invalid RSA key size value *key-size*-value that is specified for secure key generation.**

Explanation:
The key size value that is specified in the CertPlist for secure key generation is not valid. The RSA key size value for secure key generation must be at least 1024 bits and a multiple of 256.

System action:
The certificate request fails and this message is written to the PKI Services log. The request is removed from the object store.

User response:
Correct the RSA key size and resubmit the request.

IKYC090I **The number of required approvals value that is specified is not valid in certificate template, nickname: *template-nickname*.**

Explanation:
PKI Services is processing a certificate request or a Simple Certificate Enrollment Protocol (SCEP) request. The value for the required number of administrators in the named certificate template that is specified with <ADMINNUM> in the ADMINAPPROVE section or PREREGISTER section in the `pkiserv.tmpl` file or in the `pkixgen.tmpl` file is not valid.

System action
The system takes one of the following actions:

- If the value that is specified for the <ADMINNUM> tag is greater than the maximum allowed value, PKI Services uses the maximum allowed value and continues.
- If the value that is specified for the <ADMINNUM> tag is less than 1, PKI Services uses a value of 1 and continues.
- If the value that is specified for the <ADMINNUM> tag is not numeric, PKI Services uses a value of 1 and continues.

System programmer response:
Correct the number and submit new requests, if you want. However, the requests already created are not affected by the corrected value.

IKYC091I **Incorrect `DBWaitTime` value in the `pkiserv.conf` file: value**

Explanation

PKI Services is reading its configuration file, `pkiserv.conf`, to locate the value that is specified for `DBWaitTime` in the `ObjectStore` section. The value that is specified is incorrect.

System action

PKI Services uses the default waiting time, 0 minutes. It stops immediately without waiting for Db2 to resume.

System programmer response

Correct the value of the DBWaitTime and restart PKI Services if you want to configure PKI Services to wait for Db2 to resume. For more information about the DBWaitTime parameter, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC092I **Non-compliant *key-alg* key size *key-size-* specified for the FIPS level requested.**

Explanation

The key size for the given key algorithm does not meet the FIPS requirement for the level that is specified in the pkiserv.envars file. This error may be reported for either an existing public/private key used for a new or renewal certificate request, or for a request that requests PKI Services to generate the key pair.

System action

The certificate request fails and this message is written to the PKI Services log. The request is not stored in the object store.

User response

Update the request to use a FIPS-compliant key size, or change the FIPS level to allow the use of weaker keys.

IKYC093I **Signature algorithm *alg* is not compliant with the current FIPS level.**

Explanation

The PKCS10, SCEP, EST, or OCSP request contains a signature algorithm which is not FIPS-compliant with the level that is specified in the pkiserv.envars file.

System action

The PKCS10, SCEP, EST, or OCSP request fails and this message is written to the PKI Services log.

User response

Contact the PKI Services administrator for the FIPS-compliant algorithms and resubmit the request.

IKYC094I **Hash algorithm *algorithm_name* is not compliant with the current FIPS level.**

Explanation

The OCSP request uses a hash algorithm which is not compliant with the FIPS level that is specified in the pkiserv.envars file.

System action

The OCSP request fails and this message is written to the PKI Services log.

User response

Contact the PKI Services administrator for the FIPS-compliant algorithms and resubmit the request.

IKYC095I **Non-compliant key type *algorithm* specified for the FIPS level requested.**

Explanation

This message is issued for one of the following cases:

1. The PKI Services CA certificate has a key type that is not compliant with the FIPS level that is specified in the pkiserv.envars file (example: a Brainpool ECC key).
2. A PKCS10, SCEP, EST, or OCSP request received by PKI Services uses a key type that is not compliant with the FIPS level that is specified in the pkiserv.envars file.

System action

This message is written to the PKI Services log.

- For case 1, PKI Services fails to start.
- For case 2, the PKCS10, SCEP, EST, or OCSP request fails.

System programmer response

If PKI Services fails to start, change the FIPS level to allow the use of weaker keys or change to a CA certificate that complies with the FIPS level that is specified in the pkiserv.envars file.

User response

Contact the PKI Services administrator for the FIPS-compliant key type and resubmit the request.

IKYC096I **Bad DBVersion in pkiserv.conf file: *value***

Explanation

The value that is specified for DBVersion in the ObjectStore section of the pkiserv.conf file is incorrect. The value of either 0 or 1 is accepted.

System action

PKI Services stops.

System programmer response

Correct the value of `DBVersion` and restart PKI Services. For more information about the `DBVersion` parameter, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC097I **`DBVersion` *value* disagrees with ObjectStore and ICL version *version***

Explanation

PKI Services determines that the version does not match the specified `DBVersion` value in the ObjectStore section in the configuration file, `pkiserv.conf`.

System action

PKI Services stops.

System programmer response

Correct the value of `DBVersion` and restart PKI Services. For more information about the `DBVersion` parameter, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC098I **Unmatched template nickname for EST processing stored in preregistration record: *template_nickname,* ESTTemplate:*value***

Explanation

When PKI Services is processing an Enrollment over Secure Transport (EST) request from a preregistered EST client, the named certificate template stored in the preregistration record is different from that specified by the ESTTemplate keyword in the pkiserv.conf file. The ESTTemplate value will be used to locate the template to gather the input fields used for certificate generation.

System action

PKI Services processes the EST request with input fields that correspond to the template specified by ESTTemplate.

System programmer response

Make sure the same template name specified by ESTTemplate is used for preregistration. If you need to change its value, you may wait until all the EST preregistration records have been processed. For more information, see Chapter 16, "Using Enrollment over Secure Transport (EST)," on page 341.

IKYC801I ***nnnn* bytes of unconsumed data transferring extensions to certificate template**

Explanation

PKI Services is processing a certificate renewal request and has encountered an internal error. The error code encountered is displayed. A description of the error is also displayed, if known.

System action

The certificate renewal request is not processed.

System programmer response

Report the error to the IBM support center.

IKYC802I **Error *nnnn* {getting certificate-section from old certificate | setting certificate-section in certificate template | removing unnecessary extension from certificate template}: *error-code-description***

Explanation

PKI Services is processing a certificate renewal request and has encountered an internal error. The error code encountered is displayed. A description of the error is also displayed, if known.

System action

The certificate renewal request is not processed.

System programmer response

Report the error to the IBM support center.

IKYC901I **Error *nnnn* initializing *sub-function-name*: *error-code-description***

Explanation

PKI Services is initializing one of its sub-functions and has encountered an error. The sub-function name and error code encountered are displayed. A description of the error is also displayed, if known.

System action

PKI Services stops.

System programmer response

This message can accompany a message more specific to the sub-function that failed. Check the log for other error messages that are issued before this one, and diagnose accordingly. Restart PKI Services after making corrections. If you are unable to diagnose the error, report the error to the IBM support center.

Note: If the message indicates an LDAP error, the error code can be an OCSF error code. If you cannot find the error code that is documented in *z/OS IBM Tivoli Directory Server Administration and Use for z/OS*, look in *z/OS Open Cryptographic Services Facility Application Programming*.

IKYC902I Error initializing the configuration file

Explanation

PKI Services is reading its configuration file to locate the object identifiers defined in the **OIDs** section. Either the section is missing, or a value has an incorrect syntax.

System action

PKI Services stops.

System programmer response

The OID values must be defined in dotted decimal form, for example:

```
sha-1WithRSAEncryption=1.2.840.113549.1.1.5
```

Correct the configuration file, and restart PKI Services. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYC903I Error *nnnn* adding CA certificate to ICL: *error-code-description*

Explanation

PKI Services is initializing and is attempting to store its own certificate authority certificate in the issued certificate list (ICL). The attempt was not successful. The error code encountered is displayed. A description of the error is also displayed, if known.

System action

PKI Services stops.

System programmer response

This message can accompany a more specific error message. Check the log for other error messages that are issued before this one and diagnose accordingly. Restart PKI Services after making corrections. If you are unable to diagnose the error, report the error to the IBM support center.

IKYC904I ICL version *version* does not match Object Store version *version*

Explanation

During PKI Services initialization, the versions of the Object Store (OST) and the Issued Certificate List (ICL) are checked to see if they match. This message is issued when they do not match.

System action

PKI Services stops.

System programmer response

This message can accompany a more specific error message. Check the log for other error messages that are issued before this one and diagnose accordingly. Restart PKI Services after making corrections. If you are unable to diagnose the error, report the error to the IBM support center.

IKYC905I Error *error-code* querying version of [`ObjectStore` | `ICL`]

Explanation

PKI Services is initializing and is attempting to query the version of the Object Store (OST) or the Issued Certificate List (ICL). The query fails. This error code is displayed.

System action

PKI Services stops.

System programmer response

This message can accompany a more specific error message. Check the log for other error messages that are issued before this one and diagnose accordingly. Restart PKI Services after making corrections. If you are unable to diagnose the error, report the error to the IBM support center.

IKYD001I Unable to open VSAM data set *data-set-name*

Explanation

PKI Services is attempting to open one of the VSAM data sets specified in the **ObjectStore** section of the `pkiserv.conf` file or its default data set name. The open has failed. The data set name is displayed.

System action

PKI Services stops.

System programmer response

Locate the failing DSN value in the `pkiserv.conf` file. Make sure that the value specifies the correct VSAM data set name and that the data set has been created. If the data set name is not specified in the `pkiserv.conf` file, then PKI Services uses the default name for the data set. Make sure that this data set exists or add the appropriate DSN value to the `pkiserv.conf` file to specify the correct data set. If migrating from a previous release of PKI Services, make sure that the additional VSAM alternate index data sets have been created properly.

If no errors are found, contact your RACF administrator to ensure that the user ID assigned to the PKI Services daemon has permission to open the data set for update. Once corrected, restart PKI Services. For information about the values specified in the PKI Services configuration file (`pkiserv.conf`), including their defaults, see "(Optional) Steps for updating the configuration file" on page 74. See also "Steps for creating the VSAM object store and ICL data sets and indexes" on page 118.

IKYD002I **DB2® RRSAF failure: Command *rrsaf_cmd* failed with return code *0xnnnn*, reason code *0xnnnn***

Explanation

A failure condition was detected when PKI Services issued Resource Recovery Services Attachment Facility (RRSAF) commands to the Db2 for z/OS database service. *rrsaf_cmd* is the RRSAF subcommand that detected the condition. This failure prevents PKI Services from using Db2 for z/OS as its storage mechanism.

System action

PKI Services stops.

System programmer response

Look up the return code and reason code that is given in this message in *Db2 for z/OS Codes* and perform any problem determination or problem resolution instructions.

IKYD003I **DB2 SQL failure: Instruction *SQL_instruction* failed with SQLCODE *sqlcode*, SQLSTATE *sqlstate***

Explanation

A failure condition was detected when PKI Services issued Structured Query Language (SQL) instructions to the Db2 for z/OS database service. *SQL_instruction* is the SQL instruction that detected the condition.

System action

PKI Services stops.

System programmer response

Look up the SQLCODE and SQLSTATE values in this message in *Db2 for z/OS Codes* and perform any problem determination or problem resolution instructions.

IKYI001I **Request denied by installation exit. RC = *nn***

Explanation

A user is requesting PKI Services. The `PKIServ` web application called an installation-provided exit program. The exit program has determined that the request should be denied. The return code from the exit program is displayed in the message.

System action

The request in not performed.

User response

Contact your web administrator.

Web administrator response

Determine why the exit program denied the request and correct the program if necessary.

IKYI002I **SAF Service IRRSPX00 Returned SAF RC = *nn* RACF RC = *nn* RACF RSN = *nn* {*diagnostic-information*}**

Explanation

A user is requesting PKI Services. The `PKIServ` web application called the IRRSPX00 SAF callable service as requested. The service was unsuccessful. The diagnostic information that follows the message describes the problem in greater detail.

The text items listed here comprise all of the possible values for *diagnostic-information* in this message and in message IKYU002I.

1
Incorrect field name specified in CertPlist: *<field-name>*.

2
<field-name> has an incorrect value.

3
Required field *<field-name>* missing from the request.

4
Request denied, not authorized.

5
Certificate generation provider is not available [for CA domain *<CA-domain-name>*].

6
Certificate generation provider indicated the following error: *<provider-specific-error-msg>*.

7
Incorrect CertId PassPhrase specified.

8
Request has been rejected by the administrator.

9
Request is still pending approval or yet to be issued.

10
Incorrect certificate specified.

11
The certificate could not be {renewed **|** revoked} because of a state change.

12
Incorrect {CertId **|** Serial Number} specified.

13
The status of the {request **|** certificate} has been changed by another process.

14
{CertIds **|** SerialNums} has an incorrect length.

15
CertAnchor area missing.

16
CertAnchor area too small.

17
CertPlist has an incorrect length.

18
CertPlist DiagInfo field missing or has an incorrect length.

19
Conflicting field names specified in CertPlist : *field-name*.

20
Incorrect action specified.

21
Incorrect status criteria specified.

22
Incorrect transaction ID specified.

23
Incorrect reason specified.

24
Incorrect SerialNum specified.

25
SerialNums has an incorrect length.

26
Summary list or CertPlist area missing.

27
Summary list or CertPlist area too small.

28
A parameter list error has been detected.

29
An internal error has occurred during RACF processing.

30
Unable to establish recovery environment.

31
Function code specified is not defined.

32
Parameter list version specified is not supported.

33
RACF not installed.

34
Certificate generation provider internal error.

35
Unexpected error.

36
Incorrect value that is specified for CA domain.

37
Client already preregistered.

38
The ReadyMessageForm or the RecoverForm is not set up correctly. The ReadyMessageForm is required to request a certificate. The RecoverForm is required to recover a certificate whose keys were generated by PKI Services.

39
The email containing the transaction ID link to pick up the certificate was not sent successfully. The requester needs to contact the administrator.

40
The email containing the key ID link to recover the certificate was not sent successfully. The recovery process stops.

41
The requester’s email address for the certificate could not be modified because the key is not generated by PKI Services.

42
: The certificate could not be renewed because the requester’s email address has been changed.

43
: The certificate could not be deleted from the token data set (TKDS) although it was deleted from the issued certificate list (ICL).

System action

The request is not performed.

User response

Correct the problem if applicable. If you cannot correct the problem, contact your web administrator.

For problem 9, try to retrieve your certificate again later. The amount of time you need to wait depends on your PKI Services operating procedures and settings. If you continue to get this message, contact your PKI Services administrator.

Web administrator response

Problems 1, 2, and 3 probably indicate an error with the certificate template. Change the certificate template definition in the `pkiserv.tmpl` file to correct the error.

Problem 4 indicates the user ID assigned to the unit of work calling the `IRRSPX00` callable service is not RACF-authorized to perform the request. Determine whether the user should have access. If so, use RACF commands to permit the user ID to the required resources.

Problem 5 indicates the PKI Services daemon process has not been started. If PKI Services is configured for multiple-CA mode, then the CA domain name is displayed as part of the diagnostic information. Start the correct instance of PKI Services; then, retry the request.

For problems 6-13, 22, and 24, or for more information about any of the preceding problems, see earlier chapters in this document and *z/OS Security Server RACF Callable Services*.

For problems 14-21, 23, and 25-35, report the error to the IBM support center.

For problem 36, PKI Services is configured for multiple-CA mode, but the CA domain name as found in the URL contains characters that cannot be used as a CA domain name. Correct the value in the URL; then, retry the request.

Problems 38, 39 and 40 probably indicate an error with the value that is specified in the ReadyMessageForm or the RecoverForm in the configuration file. Change the value in the `pkiserv.conf` file to correct the error.

For problem 43, you need to remove the orphaned TKDS objects yourself; for example, by using ICSF panels.

PKI Services administrator response

For problem 9, locate the pending certificate request using the PKI Services administration web pages, and approve or reject the request.

IKYI003I **PKI Services CGI error in cgi-program-name: *diagnostic-error-information***

Explanation

A user is requesting PKI Services. The `PKIServ` web application CGI program processing the request detected a problem. The name of the CGI program and additional diagnostic information is displayed in the message.

System action

The request in not performed.

User response

Contact your web administrator.

Web administrator response

Locate the CGI program mentioned in the message. (Its default installation location is in a subdirectory under `/usr/lpp/pkiserv/PKIServ`.) Examine the CGI program's source code to determine the spot where it is failing and why. In most cases, the problem is caused by an error in the PKI Services template file (usually in `/etc/pkiserv/pkiserv.tmpl`). Correct the problem and retry the request. For more information, see Chapter 11, “Customizing the end-user web application if you use REXX CGI execs,” on page 135 and Chapter 12, “Customizing the administration web pages if you use REXX CGI execs,” on page 233.

IKYI004I **Installation exit failed. RC = *nn***

Explanation

A user is requesting PKI Services. The `PKIServ` web application called an installation-provided exit program. The exit program either terminated abnormally or returned an unsupported return code value. The return code from the invocation of the exit program is displayed in the message.

System action

The request in not performed.

User response

Contact your web administrator.

Web administrator response

Determine why the exit program has failed and correct the program as necessary.

IKYI005I **Invalid return type specified for certificate retrieval**

Explanation

A user is retrieving a certificate from the PKIServ web application. The result returned does not match the specified certificate type.

System action

The certificate is not returned through the web page.

User response

Specify the correct return type for certificate retrieval and try again. Make sure that the transaction ID specified corresponds to PKI/SAF Browser Certificate, PKI/SAF Server Certificate, or PKI Key Certificate.

IKYI006I **PKI Services JSP error in *jsp-filename*: *diagnostic-information***

Explanation

A user made a request through the PKI Services web application. The PKIServ web application detected a problem. The name of the JSP file and additional diagnostic information is displayed in the message.

System action

The request is not performed.

User response

Contact your web administrator.

Web administrator response

If you cannot determine the error and have modified the failing JSP file, try temporarily replacing the modified JSP with a copy of the original JSP file that shipped with PKI Services to determine whether your change is causing the problem.

IKYI007I **PKI Services web pages have not been configured for the client authentication needed for this function.**

Explanation

To renew or revoke a browser certificate, the WebSphere Application Server must be set up for client authentication. This step was not completed or client authentication is not working properly.

System action

The request is not performed.

User response

Contact your web administrator.

Web administrator response

Set up the WebSphere Application Server for client authentication. For instructions, see “Allowing WebSphere users to renew and revoke browser certificates” on page 247.

IKYK001I **Unexpected PKCS#11 *function-name* return code 0x*nnnn*. The request is not processed.**

Explanation

PKI Services is calling the PKCS #11 function *function-name* to perform an action on a token. *function-name* returns an unexpected hexadecimal return code, *nnnn*.

System action

The request is not processed

System programmer response

For more information, see *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*.

IKYK002I **PKCS#11 token unavailable for *function-name* with return code 0x*nnnn* . The request is not processed.**

Explanation

PKI Services is calling the PKCS #11 function *function-name* to perform an action on a token. *function-name* returns a hexadecimal return code, *nnnn*. The possible return codes and reasons are:

X'00e0'
: ICSF is not active or is not configured for PKCS #11 services.

X'00e1'
The PKI Services daemon has insufficient authority to access the token.

System action

PKI Services does not process the request.

System programmer response

Ensure that ICSF is properly configured and operational. If the return code is X'00e1', look for the ICH408I message that is issued for insufficient authority to the CRYPTOZ class resource and give the PKI Services daemon the required access. For more information, see *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*.

IKYK003I **{Certificate | key | PKCS12 package} with KEYID *keyid* is not deleted from TKDS object {Certificate object | Public key object | Private key object | Data object}.**

Explanation

During certificate deletion processing, PKI Services is calling a PKCS #11 function to delete a certificate and its related objects from the token data set (TKDS). The deletion is completed in the issued certificate list (ICL) but not in the TKDS.

System action

The certificate is deleted from the ICL but the certificate and its related objects are not deleted from the TKDS.

System programmer response

Look at message IKYK001I to find out the return code. For information about deleting objects from the TKDS, see the PKCS #11: Cryptographic Token Interface Standard (ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-11/v2-20/pkcs-11v2-20.pdf).

IKYK004I **PKI Services cannot generate certificates with secure keys.**

Explanation
PKI Services has determined that secure key generation has been requested in one of these ways:

- The `SecureKey` field in the SAF section of the configuration file `pkiserv.conf` is set.
- The system in which PKI Services is running enforces secure key generation through the security product.

However, the system is not set up to generate secure PKCS #11 keys.

System action:
The request fails.

System programmer response
Perform one of the following actions:

- Configure ICSF for secure PKCS #11 services.
- Set the `SecureKey` value to false and have your security administrator grant the PKI daemon the authority to generate clear keys.

For information about setting up secure and clear key generation on the TKDS, see *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*.

IKYL001I **Error *nnnn* {importing | converting} LDAP username *user's-distinguished-name*: *error-code-description***

Explanation

PKI Services is reading its configuration file to locate one of the values specified for `AuthName` in the **LDAP** section. The value that is specified has a syntax error. The incorrect value is displayed. A description of the error is also displayed, if known.

System action

PKI Services binds to the LDAP directory anonymously and continues processing. When PKI Services attempts to post certificates and CRLs to this directory, it can fail due to insufficient access. Look for message IKYC007I to determine this is happening. (RC = `LDAP_INSUFFICIENT_ACCESS`)

System programmer response

Locate the incorrect `AuthName` value in the `pkiserv.conf` file and correct it. The value must be specified as an LDAP distinguished name, for example, `CN=root,O=IBM`. Note: The OID qualifiers must be specified in uppercase and there cannot be any spaces surrounding the equal signs or commas separating the attribute value assertions (AVAs). Make corrections as needed, then stop and restart PKI Services. For more information, see "Steps for tailoring the LDAP section of the configuration file" on page 108.

IKYL002I **LDAP bind to *LDAP-server-domain-name:port* failed, status = nnnn: *status-code-description***

Explanation

PKI Services is attempting to bind to one of the LDAP servers specified in the **LDAP** section of the `pkiserv.conf` file. The bind has failed. The failing server name is displayed. A description of the error is also displayed, if known. Note: If the error code is an LDAP return code, no error description is displayed.

System action

PKI Services attempts to bind to your other LDAP servers, if any. If PKI Services is unable to bind to any LDAP servers, the LDAP posting of certificates and CRLs is temporarily suspended. PKI Services attempts to bind again during the next posting interval. All post requests remain in the request database to be attempted later, subject to being deleted after one week of unsuccessful attempts.

System programmer response

Diagnose the problem indicated by the return code. For `LDAP_SERVER_DOWN`, ensure that your LDAP server is running. If so, it is possible you have specified the server name incorrectly in the PKI Services configuration file. Locate the failing `Server` value in the `pkiserv.conf` file. Correct the value if it does not specify the correct LDAP server domain name and port, then stop and restart PKI Services. For all other LDAP errors, follow the instructions in *z/OS IBM Tivoli Directory Server Client Programming for z/OS*. Report errors to the IBM support center. If message IKYC009I is also displayed, report that information also. For more information, see "Steps for tailoring the LDAP section of the configuration file" on page 108.

IKYL003I Incorrect value that is specified for LDAPBIND or FACILITY Class profile *profile-name*

Explanation

PKI Services LDAP bind processing is trying to retrieve its LDAP bind information in preparation for communicating with the LDAP server. The bind information is contained in either an LDAPBIND class profile or the IRR.PROXY.DEFAULTS profile in the FACILITY class. Either the profile does not exist or some of the information is missing or incorrect. The name of the profile in question is displayed.

System action

PKI Services attempts to bind to your other LDAP servers, if any. If PKI Services is unable to bind to any LDAP servers, the LDAP posting of certificates and CRLs is temporarily suspended. PKI Services attempts to bind again during the next posting interval. All post requests remain in the request database to be attempted later, subject to being deleted after one week of unsuccessful attempts.

System programmer response

Locate the profile name in the PKI Services configuration file and correct it if needed. If you make corrections, stop and restart PKI Services. If the profile name is already correct, contact your RACF administrator. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

RACF administrator response

Display the PROXY segment of the profile using the RLIST TSO command. Check the LDAPHOST for accuracy, and correct it if needed. If non-anonymous access is required, do the same for the BINDDN and BINDPW.

Note: The BINDPW value is not displayed. Respecify it to ensure that it is accurate.

To alter the fields, use the RALTER TSO command. If the profile does not exist, create it using the RDEFINE TSO command. For more information, see *z/OS Security Server RACF Command Language Reference*.

IKYL004I Bad LDAP Server value *server-value* in pkiserv.conf file

Explanation

PKI Services LDAP bind processing is trying to retrieve its LDAP bind information in preparation for communicating with the LDAP server. (The *Server1*, *Server2*, and so forth keywords in the **LDAP** section of the PKI Services configuration file specifies the server host name information.) The host name has been specified incorrectly. Its value is displayed.

System action

PKI Services attempts to bind to your other LDAP servers, if any. If PKI Services is unable to bind to any LDAP servers, the LDAP posting of certificates and CRLs is temporarily suspended. PKI Services attempts to bind again during the next posting interval. All post requests remain in the request database to be attempted later, subject to being deleted after one week of unsuccessful attempts.

System programmer response

Locate the server name in the PKI Services configuration file, and correct it if needed. If you make corrections, stop and restart PKI Services. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYO001I **Error *nnnn* {setting | getting} *certificate-field* {in certificate | from template}: *error-code-description***

Explanation

PKI Services is processing a certificate request field and has encountered an internal error. The field name and the error code encountered are displayed. A description of the error is also displayed, if known.

System action

The certificate request is not processed.

System programmer response

Report the error to the IBM support center.

IKYO002I ***nnnn* bytes of unconsumed data transferring *certificate-field* to certificate**

Explanation

PKI Services is processing a certificate request field and has found that the field is larger than it should be. This is an internal error. The field name and the number of extra bytes are displayed.

System action

The certificate request is not processed.

System programmer response

Report the error to the IBM support center.

IKYO003I **The certificate request failed validity checks. Status is *nnnn*: *status-code-description***

Explanation

PKI Services is processing a certificate request field and has encountered an internal error. The field name and the status (error) code encountered are displayed. A description of the error is also displayed, if known.

System action

The certificate request is not processed.

System programmer response

Report the error to the IBM support center.

IKYO004I ***action-being-performed* returned *nnnn*: *error-code-description***

Explanation

PKI Services is processing a request and has encountered an internal error. The action being performed and the error code encountered are displayed. A description of the error is also displayed, if known.

System action

The request is not processed.

System programmer response

Report the error to the IBM support center.

IKYO005I **PKI CA certificate last used serial number was regressed.**

Explanation

During certificate fulfillment, PKI Services has detected that the CA certificate's last serial number has regressed to a serial number below the next available PKI Services serial number.

System action

PKI Services calls the `R_DataLib` callable service `IncSerialNum` function to increment the regressed serial number back to the next available PKI Services serial number or to one that was never used by PKI Services. If this is an asynchronous certificate request, the request is tried again with a new serial number.

IKYO006I **Unsupported character(s) in the UserNoticeText*n* value in the pkiserv.conf file**

Explanation

PKI Services is attempting to create the `CertificatePolicies` extension for a certificate with the specified `UserNoticeText`*n* value in the **CertPolicy** section of the `pkiserv.conf` file. The specified value contains one or more control characters.

System action

The request is not processed.

System programmer response

If you want the request to be processed, correct the value of UserNoticeText*n* and restart PKI Services. For more information about the `UserNoticeText`*n* keyword, see "Using certificate policies" on page 284.

IKY0007I **Request *transaction_ID* is not compliant with the current FIPS level.**

Explanation

The request with the indicated transaction ID was accepted when PKI Services was running at a lower FIPS level. Since then, the FIPS level has changed to a higher value, and the request does not comply with the requirements of the new FIPS level.

System action

PKI Services does not create a certificate for this request and this message is written to the PKI Services log. The request is not removed and remains in its current state.

System programmer response

Either change the FIPS level to the previous value and restart PKI Services or delete the request and notify the requester to resubmit the request.

IKYP001E **ICSF UNAVAILABLE. CERTIFICATE PROCESSING SUSPENDED**

Explanation

PKI Services background certificate processing is attempting to create a digital signature. ICSF manages the private key that is required for digital signing but it is not available for any of the following possible reasons:

- ICSF is inactive or incorrectly configured.
- The user ID of the PKI Services daemon has insufficient authority to use the ICSF private key.
- A system administrator inadvertently deleted the ICSF signing certificate and its private key.

After the ICSF problem has been corrected, PKI Services must be stopped and restarted.

System action

PKI Services background certificate processing is suspended. No certificates or CRLs are issued until the problem is corrected and PKI Services is stopped and restarted. However, certificate request management functions are still available through the `R_PKIServ` callable service and the PKI Services web pages.

System programmer response

Ensure that ICSF and the PCI cryptographic coprocessor (if applicable) are properly configured and operational. Follow the documentation for any issued message with the **CSF** prefix.

If ICH408I messages are issued for insufficient authority to CSFKEYS or CSFSERV class resources, then the user ID of the PKI Services daemon has insufficient authority to use the key. Give the user ID the required access to the specified resource.

To determine whether the key you are using requires the PCI cryptographic coprocessor, see Chapter 22, “RACF administration for PKI Services,” on page 481.

For more information, see “Installing and configuring ICSF” on page 33, *z/OS Cryptographic Services ICSF System Programmer's Guide*, and *z/OS Cryptographic Services ICSF Administrator's Guide*.

If you make changes to ICSF to correct the problem, stop and restart PKI Services. For more information, see “(Optional) Steps for updating the configuration file” on page 74.

Routing code

2

Descriptor code

6

IKYP002I **PKI SERVICES INITIALIZATION COMPLETE**

Explanation

PKI Services has been started and has finished initializing.

System action

PKI Services processing continues.

Routing code

2

Descriptor code

6

IKYP003I **PKI SERVICES SHUTDOWN REQUESTED**

Explanation

One of the following occurs:

- An operator command was issued to stop PKI Services.
- PKI Services detects an unrecoverable failure condition.

System action

PKI Services stops.

Routing code

2

Descriptor code

6

IKYP004I LOG OPTION PROCESSED: *log-option*

Explanation

A MODIFY operator command was issued to alter the current log setting for PKI Services.

System action

The log setting for PKI Services is changed as requested.

Routing code

2

Descriptor code

5

IKYP005I INCORRECT LOG OPTION SPECIFIED

Explanation

A MODIFY operator command was issued to alter the current log setting for PKI Services. The log parameter syntax or value is incorrect.

System action

The MODIFY command is not processed. The log setting for PKI Services is unchanged.

System programmer response

Execute the MODIFY command specifying a correct log parameter. For more information, see "Changing logging options" on page 541.

Routing code

2

Descriptor code

5

IKYP006I UNRECOGNIZED PKI SERVICES COMMAND: SPECIFY LOG, DISPLAY, OR STOP

Explanation

A MODIFY operator command was issued for PKI Services. The command specified is not a supported PKI Services command.

System action

The MODIFY command is not processed. PKI Services continues processing unchanged.

System programmer response

Execute the MODIFY command specifying a supported PKI Services command. For more information, see "Stopping the PKI Services daemon" on page 131 and "Changing logging options" on page 541.

Routing code

2

Descriptor code

5

IKYP007E INSUFFICIENT STORAGE AVAILABLE

Explanation

PKI Services is attempting to allocate storage for processing a MODIFY operator command, but is unsuccessful because of a storage shortage.

System action

The console command is not processed. However, PKI Services can continue processing normally.

Operator response

Report the problem to your system programmer. After the problem is corrected, you can execute the command again.

System programmer response

Increase the region size for the PKI Services started procedure. Stop and restart PKI Services. For more information, see "Steps for starting the PKI Services daemon" on page 129 and "Stopping the PKI Services daemon" on page 131.

Routing code

2

Descriptor code

5

IKYP008E DIRECTORY POST UNSUCCESSFUL. LDAP DATA LIBRARY MODULE RC = *nnnn*

Explanation

PKI Services background certificate processing is attempting to post information (such as a certificate or CRL) to a directory. The post was unsuccessful. The OCSF Data Library Module (LDAPDL) return code is displayed in the message.

System action

The information is not posted now. The post request remains in the PKI Services request database to be reattempted later. If posting continues to be unsuccessful for one week, the information is removed from the request database.

System programmer response

Determine the cause of the failure from the return code that is displayed and take appropriate action. These return codes are documented in *z/OS Open Cryptographic Services Facility Application Programming*. If the error is `LDAPDL_NO_SUCH_OBJECT`, the LDAP entry could not be created because the required suffix does not exist. Check the PKI Services log to determine the entry that could not be created, as indicated on messages IKYC005I and IKYC008I. If the entry should be posted to LDAP, you need to define the suffix in the LDAP server configuration file and recycle the LDAP server. For more information, see "Steps for installing and configuring LDAP" on page 31 and *z/OS IBM Tivoli Directory Server Administration and Use for z/OS*.

If you want PKI Services to bypass LDAP posting for certificates with missing suffixes, set `RetryMissingSuffix=F` in the PKI Services `pkiserv.conf` configuration file. Then, stop and restart the PKI Services daemon. For more information, see "Steps for tailoring the LDAP section of the configuration file" on page 108.

Routing code

2

Descriptor code

6

IKYP009I PKI SERVICES IS STARTING, FMID *product-fmid*

Explanation

The START operator command was issued to start PKI Services. The START command could have been entered directly at the operator's console or indirectly through a COMMND*xx* PARMLIB member.

System action

PKI Services initialization proceeds.

Routing code

2

Descriptor code

6

IKYP010I THE CONFIGURATION FILE NAME EXCEEDS THE MAXIMUM LENGTH OF *nnnn* CHARACTERS

Explanation

The PKI Services daemon process is starting. Initialization processing is reading the `_PKISERV_CONFIG_PATH` environment variable. The value that is specified is too long.

System action

PKI Services stops.

System programmer response

Determine the location of your PKI Services environment variables file, and correct the value that is specified for `_PKISERV_CONFIG_PATH`. Then, restart PKI Services.

Routing code

2

Descriptor code

6

IKYP011I PKI SERVICES ADDRESS SPACE COULD NOT BE MADE NON-SWAPPABLE: ERROR *nnnn*

Explanation

The PKI Services daemon process is starting. Initialization processing is attempting to make the PKI Services address space non-swappable. The attempt was unsuccessful. The SYSEVENT TRANSWAP error code is displayed.

System action

PKI Services stops.

System programmer response

Look up the error code for SYSEVENT TRANSWAP in *z/OS MVS Programming: Authorized Assembler Services Reference SET-WTO* to determine what to do. Then, restart PKI Services.

Routing code

2

Descriptor code

6

IKYP012I SYSTEM FUNCTION *function-name* DETECTED ERROR - *error-string*

Explanation

PKI Services processing received an error when calling a system service. The service name and error message are displayed.

System action

PKI Services stops.

System programmer response

See documentation that is related to the service that failed. Make any necessary corrections. Then, restart PKI Services.

Routing code

2

Descriptor code

6

IKYP013I PKI SERVICES DETECTED AN ERROR DURING INITIALIZATION: ERROR *nnnn*, REASON 0x*nnnn*

Explanation

PKI Services is starting. Initialization processing is attempting to set up the Program Call (PC) interface. The attempt was unsuccessful. The error and reason codes are displayed.

System action

PKI Services stops.

System programmer response

Determine the failing service by examining the error code. The values are as follows:

Note: This message is also issued after various C function calls. In these cases, ERROR is the value of `errno`, and REASON is the value of `__errno2()`.

1
The PKI Services daemon (IKYPKID) is not APF-authorized.

3
Unable to establish recovery. The reason code displayed is the ESTAEX macro return code.

5
Unable to create a PC linkage table index. The reason code displayed is the LXRES macro return code.

6
Unable to create a PC entry table. The reason code displayed is the ETCRE macro return code.

7
Unable to connect the PC entry table to the linkage table. The reason code displayed is the ETCON macro return code.

8, 10, or 11
Unable to create a name token entry. The reason code displayed is the IEANTCR callable service return code.

For error code **1**, make the IKYPKID load module in SYS1.LINKLIB APF-authorized. For all other error codes, see the documentation that is associated with the MVS service that failed. Make corrections as necessary. Then, restart PKI Services.

Routing code

2

Descriptor code

6

IKYP014I PKI Services detected an error during termination: Error *nnnn*, Reason *nnnn*

Explanation

PKI Services is stopping. Termination processing is attempting to free resources allocated. The attempt was unsuccessful. The error and reason codes are displayed.

System action

PKI Services termination processing continues.

System programmer response

PKI Services should end normally. If so, no action is needed. However, if you want to diagnose the problem, determine the failing service by examining the error code:

16
: Unable to establish recovery. The reason code displayed is the ESTAEX macro return code.

See associated documentation for the MVS service that failed. Make corrections as necessary.

IKYP015I A PKI Services program call request failed: Error *nnnn*

Explanation

PKI Services is processing a PC request. The PC request was canceled before PKI Services completed processing on it. The error code that was posted at the time of the cancel is displayed.

System action

PKI Services processing continues.

System programmer response

If the error code is 8, no action is required. This is an informational message only. For all other error codes, contact your IBM support center.

IKYP016I THE PKI SERVICES RUNTIME ENVIRONMENT COULD NOT BE INITIALIZED

Explanation

The PKI Services daemon process is starting. Initialization processing is trying to initialize the PKI Services runtime environment within the daemon address space. The attempt was unsuccessful.

System action

PKI Services stops.

System programmer response

Look for other PKI Services log messages that are related to this error. For more information, see Chapter 25, "Using information from the PKI Services logs," on page 537.

Routing code

2

Descriptor code

6

IKYP017I PKI SERVICES IS ALREADY RUNNING

Explanation

An attempt was made to start more than one instance of the PKI Services daemon.

System action

The first instance of PKI Services continues processing. The second instance stops.

Routing code

2

Descriptor code

6

IKYP018I PKI Services initialization failed because the program is not APF-authorized

Explanation

PKI Services is starting. Initialization processing is attempting to initialize the PKI Services runtime environment within the daemon address space. The attempt was unsuccessful because the PKI Services daemon (IKYPKID) is not APF-authorized.

System action

PKI Services stops.

Routing code

2

Descriptor code

6

IKYP019I PKI Services dump created.

Explanation

PKI Services encountered a severe error during processing and has dumped the process (using the CEE3DMP callable service).

System action

PKI Services processing ends.

Operator response

Contact your system programmer.

System programmer response

Examine the dump to determine the error. Contact the IBM support center if needed. After the error has been corrected, restart PKI Services. For more information, see "Steps for starting the PKI Services daemon" on page 129 and "Stopping the PKI Services daemon" on page 131.

IKYP020I PKI SERVICES RESTART REGISTRATION COMPLETE ON *system-name*

Explanation

PKI Services is starting. Initialization processing has successfully registered PKI Services for automatic restart (ARM).

System action

PKI Services processing continues.

Routing code

2

Descriptor code

6

IKYP021I PKI SERVICES RESTARTING ON *system-name*

Explanation

The PKI Services daemon stopped and is being restarted by the Automatic Restart Manager (ARM). The restart was successful.

System action

PKI Services processing continues.

Routing code

2

Descriptor code

6

IKYP022I UNABLE TO REGISTER PKI SERVICES FOR RESTART: ERROR *nnnn*, REASON 0x*nnnn*

Explanation

PKI Services is starting. Initialization processing is attempting to register PKI Services for automatic restart (ARM), using the IXCARM macro service. The attempt was unsuccessful. The IXCARM return and reason codes are displayed. Note: The reason code is displayed in hexadecimal.

System action

PKI Services initialization continues without automatic restart capability.

System programmer response

Determine and correct the problem with IXCARM as indicated by the error codes displayed. Then, stop and restart PKI Services if you want automatic restart capability. For more information, see *z/OS MVS Programming: Sysplex Services Reference*.

Routing code

2

Descriptor code

6

IKYP023I PKI Services failed to format the display message

Explanation

A MODIFY operator command was issued to display the current settings for PKI Services. Formatting of the display information failed.

System action

The settings are not displayed. PKI Services processing continues.

System programmer response

Report the error to the IBM support center.

IKYP024I PKI SERVICES DUMPING FOR ABEND *abend-code* RC *nnnn*

Explanation

PKI Services has incurred an abend. The abend and reason codes are displayed.

System action

PKI Services stops.

System programmer response

Use IPCS to examine the dump and diagnose the problem. Contact the IBM support center if necessary. Restart PKI Services after the error has been corrected.

Routing code

2

Descriptor code

6

IKYP025I PKI SERVICES SETTINGS:

Explanation

A MODIFY operator command was issued to display the current settings for PKI Services.

System action

The settings are displayed.

```
IKYP025I PKI SERVICES SETTINGS:
   CA DOMAIN NAME: {CA-domain-name-for-this-PKI-daemon}
   SUBCOMPONENT              MESSAGE LEVEL
     LDAP                    {current-message-level}
     SAF                     {current-message-level}
     DB                      {current-message-level}
     CORE                    {current-message-level}
     PKID                    {current-message-level}
     POLICY                  {current-message-level}
     TPOLICY                 {current-message-level}
   MESSAGE LOGGING SETTING: {STDERR_LOGGING |
 STDOUT_LOGGING}
   CONFIGURATION FILE IN USE:
 {full-UNIX-pathname-of-configuration-file-being-used}
   TEMPLATE FILE IN USE:
 {full-UNIX-pathname-of-template-file-being-used}
   CA CERTIFICATE FINGERPRINTS:
      SHA1:   {EBCDIC-representation-of-sha1-hash}
      MD5:    {EBCDIC-representation-of-md5-hash}
      SHA256: {EBCDIC-representation-of-sha256-hash}
      SHA512: {EBCDIC-representation-of-sha512-hash}
   FIPS LEVEL: {fips-level}
   STATUS: {OPERATIONAL | PAUSED}
```

Restrictions:

- The `CA DOMAIN NAME:` line is suppressed if the daemon is running in single CA mode.
- For long CA domain names, the *`CA-domain-name-for-this-PKI-daemon`* value is truncated to 50 characters.
- The `CA CERTIFICATE FINGERPRINTS:` line that shows the MD5 value is suppressed if the *fips-level* value is not NONE.

The possible *current-message-level* values for each subcomponent are:

- SEVERE MESSAGES ONLY
- ERROR MESSAGES AND HIGHER
- WARNING MESSAGES AND HIGHER
- INFORMATIONAL MESSAGES AND HIGHER
- DIAGNOSTIC MESSAGES AND HIGHER
- VERBOSE DIAGNOSTIC MESSAGES AND HIGHER

The possible *fips-level* values are:

- NONE
- FIPS 140-2
- SP800-131A WITH EXCEPTION
- SP800-131A WITHOUT EXCEPTION

Operator response

You can change the subcomponent message levels with the MODIFY operator command if you want. For more information, see "Changing logging options" on page 541.

Routing code

2

Descriptor code

5

IKYP026E PKI SERVICES {CA | RA} CERTIFICATE EXPIRES ON *yyyy/mm/dd*

Explanation

The certificate that contains the PKI Services CA or RA public key expires on the date shown.

System action

If the certificate has not yet expired, processing continues as normal. After the CA certificate expires, certificates issued by PKI Services can be unusable depending on their usage.

System programmer response

You should renew the certificate before it expires. If your security product is RACF, your certificate is contained in a RACF profile established when you first configured PKI Services. Follow RACF documentation on how to renew a certificate. This is done using either the RACDCERT TSO command or RACF ISPF panels. For more information, see "Renewing your PKI Services CA and RA certificates" on page 488 and *z/OS Security Server RACF Security Administrator's Guide*.

Routing code

2

Descriptor code

6

IKYP027E ERROR ACCESSING PKI SERVICES CA CERTIFICATE

Explanation

The PKI Services CA certificate is stored in the security product's database. PKI Services background certificate processing is attempting to access the certificate using the `R_datalib` SAF callable service. The attempt failed. Message IKYS015I should also appear in the PKI Services log.

System action

PKI Services background certificate processing is suspended. No certificates are issued until the problem is corrected. However, certificate request management functions are still available through the `R_PKIServ` callable service and the PKI Services web pages.

System programmer response

You need to determine why the access failed. Look up the `R_datalib` return code that is displayed on message IKYS015I in *z/OS Security Server RACF Callable Services*. If your security product is RACF, your certificate is contained in a RACF profile that is established when you first configured PKI Services. That certificate must be connected as the default certificate to the key ring identified by the KeyRing keyword in the PKI Services configuration file. (The default location for this file is `/etc/pkiserv/pkiserv.conf`.) If you have only renewed your certificate and have not recycled PKI Services, stopping and restarting the PKI Services daemon can solve the problem. If not, use the RACF RACDCERT LIST and LISTRING commands to determine whether the correct certificate is connected to the key ring. Also, use the RACF RLIST command to check that the PKI Services daemon user ID has proper authority to access the profile. Make any required changes. Then, stop and restart PKI Services. For more information, see Chapter 22, "RACF administration for PKI Services," on page 481 and *z/OS Security Server RACF Security Administrator's Guide*.

Routing code

2

Descriptor code

6

IKYP028E PKI SERVICES DISTINGUISHED NAME OR KEY CHANGE ERROR

Explanation

PKI Services is starting. Initialization processing has retrieved the PKI Services signing certificate from the key ring assigned to PKI Services. The certificate is incompatible with certificate processing that has previously transpired. The subject's distinguished name or the public key or both differ from the previous values used. The subject's distinguished name cannot be changed without reconfiguring PKI Services. The public key can be changed, but only if the key rollover process is performed.

System action

PKI Services stops.

System programmer response

Determine whether PKI Services is processing the correct certificate. If your security product is RACF, your certificate is contained in a RACF profile that is established when you first configured PKI Services. That certificate must be connected as the default certificate to the kePKI Servicesy ring identified by the KeyRing keyword in the configuration file. (The default location for this file is `/etc/pkiserv/pkiserv.conf`.) Use the RACF RACDCERT LIST and LISTRING commands to determine whether the correct certificate is connected to the key ring. If you are attempting to rekey the PKI Services CA, you must follow the rollover process that is detailed in Chapter 22, "RACF administration for PKI Services," on page 481. Make any required changes. Then, restart PKI Services. For more information, see *z/OS Security Server RACF Security Administrator's Guide*.

Routing code

2

Descriptor code

6

IKYP029I PKI Services can only be started from a started procedure

Explanation

An attempt that is made at starting the PKI Services daemon was rejected because it was not made from a started procedure.

System action

The PKI Services daemon halts its initialization and stops after displaying this message to the standard output (STDOUT) of the process.

System programmer response

Use the started procedure that PKI Services supplies in SYS1.PROCLIB(PKISERVD). For more information, see "Steps for starting the PKI Services daemon" on page 129.

IKYP030I CRL APPROACHING MAXIMUM SIZE

Explanation

PKI Services is creating CRLs as part of CRL processing and has encountered at least one CRL that is approaching the maximum size for CRL posting objects in the object store. This can occur when large CRL posting has not been configured.

System action

PKI Services CRL processing continues. If the CRLs are all less than the record size limit of approximately 32 K bytes, CRL processing within PKI Services functions normally. However, CRL processing outside of PKI Services can be adversely affected due to the size of the CRL. If any CRL exceeds the record size limit, PKI Services CRL processing is unsuccessful, and the large CRLs are not published to the LDAP directory. When this happens you also receive message IKYC010I with the error code description, `Record too long`.

System programmer response

It is imperative that you correct the situation immediately. You can take either of these approaches:

- If you want to continue to use VSAM records or Db2 tables for LDAP posting, and if you are not yet using distribution point CRLs, start using them now. Edit the PKI Services configuration file and add the `CRLDistSize` directive to the **CertPolicy** section. If you are already using distribution point CRLs, decrease the value that is specified for the `CRLDistSize` directive. Make the appropriate changes and save the configuration file.

  Note: These changes do not result in an immediate reduction in the size of the CRL. You continue to see this message until the revoked certificates on the CRL expire and are removed from the CRL.
- Alternatively, you can enable large CRL posting. If you do this, PKI Services stores CRLs in a z/OS UNIX file system instead of in a VSAM data set or Db2 table, and the record size limit of approximately 32 K bytes does not apply. Edit the PKI Services configuration file and add the `EnableLargeCRLPosting` and `LargeCRLPath` directives to the **CertPolicy** section. In addition, you need to configure a z/OS UNIX file system to hold CRLs. For more information, see "Enabling support for large CRLs" on page 297.

Guideline: Enable large CRL posting.

When the configuration file is saved, stop and restart PKI Services. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYP031E [RSA | DSA | ECC] signing key algorithm error

Explanation

PKI Services is reading the **CertPolicy** section of its configuration file (`pkiserv.conf`) to find the signing algorithm. One of the following conditions occurred:

- The CA certificate key type does not match, or its key size is incompatible with the signature algorithm that you specified with the `SigAlg1` value in the **CertPolicy** section of the `pkiserv.conf` configuration file.
- The OID corresponding to the specified algorithm in the **OIDs** section is incorrect or is not specified at all.

System action

PKI Services stops.

System programmer response

Make sure the `SigAlg1` value in the **CertPolicy** section and its corresponding OID value in the **OIDs** section are correct and compatible with the CA certificate's key type.

If the CA certificate key type is RSA, specify the `SigAlg1` algorithm value as one of the following:

- `sha-1WithRSAEncryption` (OID value `1.2.840.113549.1.1.5`)
- `sha-256WithRSAEncryption` (OID value `1.2.840.113549.1.1.11`)
- `sha-384WithRSAEncryption` (OID value `1.2.840.113549.1.1.12`)
- `sha-512WithRSAEncryption` (OID value `1.2.840.113549.1.1.13`)

- `sha-224WithRSAEncryption` (OID value `1.2.840.113549.1.1.14`)
- `md-5WithRSAEncryption` (OID value `1.2.840.113549.1.1.4`)
- `md-2WithRSAEncryption` (OID value `1.2.840.113549.1.1.2`)
- `rsassa-pss,sha-256Hash` (OID value pair `1.2.840.113549.1.1.10, 2.16.840.1.101.3.4.2.1`)
- `rsassa-pss,sha-384Hash` (OID value pair `1.2.840.113549.1.1.10, 2.16.840.1.101.3.4.2.2`)
- `rsassa-pss,sha-512Hash` (OID value pair `1.2.840.113549.1.1.10, 2.16.840.1.101.3.4.2.3`)

For any `rsassa-pss` algorithms, the CA key size must be at least 2048 bits.

If the CA certificate key type is DSA, specify the `SigAlg1` algorithm value as follows:

- `id-dsa-with-sha1` (OID value `1.2.840.10040.4.3`)

If the CA certificate key type is ECC, specify the `SigAlg1` algorithm value as follows:

- `ecdsa-with-sha1` (OID value `1.2.840.10045.4.1`)
- `ecdsa-with-sha224` (OID value `1.2.840.10045.4.3.1`)
- `ecdsa-with-sha256` (OID value `1.2.840.10045.4.3.2`)
- `ecdsa-with-sha384` (OID value `1.2.840.10045.4.3.3`)
- `ecdsa-with-sha512` (OID value `1.2.840.10045.4.3.4`)

Correct the configuration values, and restart PKI Services. For more information, see "Updating the signature algorithm" on page 287.

IKYP032I PKI SERVICES DOES NOT HAVE RA CAPABILITY. SCEP PROCESSING SUSPENDED

Explanation

The PKI Services daemon process is starting. Initialization processing determines that the SCEP interface should be enabled and it reads the contents of the key ring to locate the certificate and key to be used for the SCEP registration authority (RA) function. No RA-capable certificate and key was found.

System action

Initialization continues but PKI Services SCEP processing is suspended.

System programmer response

The RA function of PKI Services requires a certificate and a private key capable of creating general purpose digital signatures and enciphering session keys, with `key usage`, `digitalSignature`, and `keyEncipherment` (`Handshaking`). Either the PKI Services CA certificate must have this capability (which is atypical) or an additional, dedicated RA certificate for PKI Services must be established. In either case, the certificate must have the proper key usage and must have an RSA private key. If a dedicated RA certificate is used, specify its label using the `RALabel` directive in the **SAF** section of the PKI Services configuration file (`pkiserv.conf`). The RA certificate must be assigned to the user ID of the PKI Services daemon and must be connected to the PKI Services key ring with USAGE PERSONAL and DEFAULT NO. For more information, see Chapter 15, "Enabling Simple Certificate Enrollment Protocol (SCEP)," on page 333. If you make changes to the PKI Services key ring to correct the problem, stop and restart PKI Services.

Routing code

2

Descriptor code

6

IKYP033I Incorrect value that is specified for CA domain name

Explanation

The PKI Services daemon process is starting. Initialization processing has determined that the CA domain name value that is specified in the `_PKISERV_CA_DOMAIN` environment variable in the `pkiserv.envars` file contains illegal characters.

System action

PKI Services stops.

System programmer response

The first 8 characters of the CA domain name are limited to the following character set: alphanumeric characters (`a-z`, `A-Z`, `0-9`) and the hyphen (`-`). In addition, the first character must not be a number or hyphen. Edit the PKI Services environment variables file `pkiserv.envars` for the CA instance in error and correct the value that is specified for `_PKISERV_CA_DOMAIN`. Restart PKI Services. For more information, see "Adding a new CA domain" on page 302.

IKYP034E ICSF UNAVAILABLE. SCEP PROCESSING SUSPENDED

Explanation

PKI Services is attempting to decrypt a Simple Certificate Enrollment Protocol (SCEP) request received from a SCEP client or to sign its response. ICSF manages the private key that is required for SCEP decryption and signing but ICSF is unavailable for any of the following possible reasons:

- ICSF is inactive or incorrectly configured.
- The user ID of the PKI Services daemon has insufficient authority to use the ICSF private key.
- A system administrator inadvertently deleted the certificate and its ICSF private key.

System action

PKI Services rejects the SCEP request.

System programmer response

Ensure that ICSF and the PCI cryptographic coprocessor (if applicable) are properly configured and operational. Follow the documentation pertaining to any issued messages having the **CSF** prefix. If you make changes to ICSF to correct the problem, stop and restart PKI Services.

If ICH408I messages are issued for insufficient authority to CSFKEYS or CSFSERV class resources, then the user ID of the PKI Services daemon has insufficient authority to use the private key. Give the user ID the required access to the specified resource.

To determine whether your key still exists or requires the PCI cryptographic coprocessor, see Chapter 22, "RACF administration for PKI Services," on page 481. To determine whether your SCEP configuration requires a CA certificate or a CA/RA combination, see "Installing and configuring ICSF" on page 33.

Routing code

2

Descriptor code

6

IKYP035I Unsupported character(s) in CA's name

Explanation

The PKI Services daemon process is starting. Initialization processing has determined that the CA's distinguished name contains one or more characters that have UTF-8 or BMP encoding that does not map to code page IBM-1047.

System action

PKI Services stops.

System programmer response

The characters in the distinguished name in the CA certificate must map to code page IBM-1047. Choose another CA certificate whose name has only characters that map to code page IBM-1047.

IKYP036I UNSUPPORTED CHARACTER(S) in RA'S NAME. SCEP PROCESSING SUSPENDED

Explanation

The PKI Services daemon process is starting. Initialization processing has determined that the RA's distinguished name contains one or more characters that have UTF-8 or BMP encoding that does not map to code page IBM-1047.

System action

Initialization continues, but PKI Services SCEP processing is suspended.

System programmer response

If SCEP support is needed, choose another RA certificate in which the name has only characters that map to code page IBM-1047.

Routing code

2

Descriptor code

6

IKYP037I ONE OR MORE AUTOMATICALLY RENEWED CERTIFICATES cannot BE SENT

Explanation

PKI Services attempted to create or retrieve the notes with the renewed certificates. An internal error occurred.

System action

PKI Services continues with the automatic renewal processing.

System programmer response

Check for more details in IKYC069I messages in the PKI Services log file and manually send the transaction IDs to the email addresses to tell the users to pick up the renewed certificates.

Routing code

2

Descriptor code

6

IKYP038I THE DIRECTORY OR FILE SPECIFIED EXCEEDS THE MAXIMUM LENGTH OF *nnn* CHARACTERS

Explanation

The PKI Services daemon process is starting. Initialization processing has determined that the value that is specified by the _PKISERV_VARDIR variable or the _PKISERV_EXIT variable in the pkiserv.envars file exceeds the limit indicated in the message.

System action

PKI Services stops.

System programmer response

Correct the name specified by _PKISERV_VARDIR or _PKISERV_EXIT and restart PKI Services.

Routing code

2

Descriptor code

6

IKYP039E DIRECTORY POST UNSUCCESSFUL. ERROR CODE = *nnnn*

Explanation

PKI Services background certificate processing is attempting to post information (such as a certificate or CRL) to a directory. The post was unsuccessful. The return code from the associated LDAP client API is displayed in the message.

System action

The information is not posted now. The post request remains in the PKI Services request database to be reattempted later. If posting continues to be unsuccessful for one week, the information is removed from the request database.

System programmer response

Determine the cause of the failure from the return code that is displayed and take appropriate action. The return codes are documented in *z/OS IBM Tivoli Directory Server Client Programming for z/OS*. If the error is LDAP_NO_SUCH_OBJECT, the LDAP entry could not be created because the required suffix does not exist. Check the PKI Services log to determine the entry that could not be created, as indicated by messages IKYC005I and IKYC008I. If the entry should be posted to LDAP, you need to define the suffix in the LDAP server configuration file and recycle the LDAP server.

For more information, see "Installing and configuring LDAP" on page 31 and *z/OS IBM Tivoli Directory Server Administration and Use for z/OS*. If you want PKI Services to bypass LDAP posting for certificates with missing suffixes, set `RetryMissingSuffix=F` in the PKI Services `pkiserv.conf` configuration file. Then, stop and restart the PKI Services daemon. For more information, see "Steps for tailoring the LDAP section of the configuration file" on page 108.

Routing code

2

Descriptor code

6

IKYP040I PKI SERVICES DOES NOT HAVE KEY GENERATION CAPABILITY

Explanation

The PKI Services daemon process is starting. Initialization processing has found that the `TokenName` field of the **SAF section** of the configuration file `pkiserv.conf` is not specified. Or `TokenName` field is specified, but an error has occurred in locating or creating the token.

System action

Initialization continues but PKI Services does not have key generation capability.

System programmer response

If you want to enable the key generation capability, make sure that the token data set (TKDS) has been set up and the system has the required hardware, and specify the `TokenName` in the `pkiserv.conf` file. For information about setting up the TKDS, see *z/OS Cryptographic Services ICSF Writing PKCS #11 Applications*.

Routing code

2

Descriptor code

6

IKYP041E PKI SERVICES CA CERTIFICATE HAS *nnnn* SERIAL NUMBERS REMAINING.

Explanation

The PKI Services CA certificate can sign a finite number of certificates before it exceeds the maximum unique serial number usable by PKI Services (X'FFFFFFFE', or 4294967294). PKI Services has detected that the CA certificate soon runs out of unique serial numbers.

System action

PKI Services processing continues.

System programmer response

Add a new CA domain to continue certificate generation capability under a new CA certificate. For more information, see "Adding a new CA domain" on page 302.

Routing code

2

Descriptor code

6

IKYP042E PKI SERVICES HAS NO REMAINING SERIAL NUMBERS FOR CERTIFICATE GENERATION.

Explanation

The PKI Services CA certificate can sign a finite number of certificates before it exceeds the maximum unique serial number usable by PKI Services (X'FFFFFFFE', or 4294967294). PKI Services has detected that the CA certificate has exceeded this maximum serial number.

System action

PKI Services background certificate processing is suspended. No certificates are issued until the problem is corrected. However, certificate request management functions are still available through the `R_PKIServ` callable service and the PKI Services web pages.

System programmer response

Add a new CA domain to continue certificate generation capability under a new CA certificate. For more information, see "Adding a new CA domain" on page 302.

Routing code

2

Descriptor code

6

IKYP043I PKI Services CA certificate cannot be a Diffie-Hellman certificate.

Explanation

An elliptic curve cryptography (ECC) certificate with only the keyAgreement keyUsage bit set, or with the keyAgreement bit set with either encipherOnly or decipherOnly set, is an ECC Diffie-Hellman certificate. Its intended usage is for key exchange, not for signing. Therefore, a CA certificate cannot be of this type.

System action

PKI Services stops.

System programmer response

Make sure that the key ring specified in the **SAF** section of the PKI Services configuration file (`pkiserv.conf`) contains a CA certificate that is not an ECC Diffie-Hellman certificate.

IKYP044I CRL NUMBER *crl-serial-number* PROCESSING {FOR CA DOMAIN *cadomain*} COMPLETED SUCCESSFULLY

Explanation:
The certificate revocation list (CRL) processing completed successfully. This message is for notification purposes.

System action:

None.

System programmer response:
None.

Routing code:
2

Descriptor code:
6

IKYP045I CRL NUMBER *crl-serial-number* PROCESSING {FOR CA DOMAIN *cadomain*} FAILED

Explanation:
The certificate revocation list (CRL) processing failed. This message is for notification purposes.

System action:
None.

System programmer response:
Look at the PKI Services job log to investigate why the CRL process failed.

Routing code:
2

Descriptor code:
6

IKYP046I PERFORMANCE OF ADMINISTRATIVE FUNCTIONS MIGHT BE DEGRADED. SAF RC *nn*, RACF RC *nn*, RACF RSN *nn*

Explanation:
During PKI Services initialization, a call to the SAF IRRSIA00 (initACEE) callable service attempted to establish a thread-level or task-level ACEE. The attempt failed with the specified return and reason codes, displayed in decimal. The call to IRRSIA00 is attempted only when the `AdminGranularControl` option is set in the `pkiserv.conf` file. The performance of administrative query and modification functions can be degraded.

System action:
Initialization continues with granular administration enabled, but the administrative functions can require more time and system resources to complete than they require with granular administration disabled.

Programmer response:
For more information about the indicated return and reason codes for IRRSIA00 (initACEE), see *z/OS Security Server RACF Callable Services*. Diagnose and correct the problem. Restart PKI Services.

Routing code:
9

Descriptor code:
6

IKYP047I PKI SERVICES DETECTS DB2 IS UNAVAILABLE

Explanation

The PKI Services daemon has detected that Db2 is not available and is waiting for Db2 to resume operation.

System action

PKI Services cannot perform any of its functions until Db2 is available.

System programmer response

Restart Db2.

Routing code:
2

Descriptor code:
6

IKYP048I PKI SERVICES DETECTS DB2 IS AVAILABLE

Explanation

The PKI Services daemon has detected that Db2 is available. The PKI Services daemon has resumed operation.

System action

PKI Services resumes operation.

System programmer response

None.

Routing code:
2

Descriptor code:
6

IKYP049I PKI SERVICES STOPPING DUE TO DB2 UNAVAILABILITY

Explanation

The Db2 subsystem that is used by PKI Services became unavailable while PKI Services was operational, and the Db2 subsystem did not resume operation within the time specified by the `DBWaitTime` option in the `pkiserv.conf` configuration file.

System action

PKI Services stops.

System programmer response

Restart PKI Services after Db2 becomes available. You may increase the time value that is specified for the `DBWaitTime` option in the `pkiserv.conf` configuration file if you want to keep PKI Services in the wait state for a longer period in the future.

Routing code:
2

Descriptor code:
6

IKYP050I PKI SERVICES COULD NOT START BECAUSE ICSF IS UNAVAILABLE

Explanation

During PKI Services initialization, attempts to use ICSF PKCS#11 services failed.

System action

PKI Services stops.

System programmer response

Ensure that ICSF is started with PKCS#11 services enabled before starting PKI Services.

Routing code

2

Descriptor code

6

IKYP051I INCORRECT _PKISERV_FIPS_LEVEL VALUE

Explanation

PKI Services is reading its environment variables file, `pkiserv.envars`, to locate the value that is specified for _PKISERV_FIPS_LEVEL. The value that is specified is incorrect.

The value of _PKISERV_FIPS_LEVEL must be one of the following:

0: non FIPS mode - this is the default
1: FIPS 140-2 (key strength 80 bits)
2: SP800-131A with exception (key strength 112 bits, legacy use of keys can still be 80 bits)
3: SP800-131A without exception (key strength 112 bits and higher, for all keys used for all operations)

System action

PKI Services stops.

System programmer response

Correct the value of _PKISERV_FIPS_LEVEL and restart PKI Services. For more information about the _PKISERV_FIPS_LEVEL parameter, see "(Optional) Steps for updating PKI Services environment variables" on page 72.

Routing code

2

Descriptor code

6

IKYP052I CA SIGNING ALGORITHM IS NOT FIPS COMPLIANT

Explanation

During PKI Services initialization, the CA signing algorithm specified in the `pkiserv.conf` file does not meet the FIPS requirement for the level that is specified in the `pkiserv.envars` file.

System action

PKI Services stops.

System programmer response

Either change the signing algorithm in the `pkiserv.conf` file or the FIPS level in the `pkiserv.envars` file to match each other.

Routing code

2

Descriptor code

6

IKYP053I CA CERTIFICATE IS NOT FIPS COMPLIANT

Explanation

During PKI Services initialization, the CA certificate is found not complying with one or more of the following conditions.

- A _PKISERV_FIPS_LEVEL other than 0 was specified and the CA certificate key is stored in ICSF Public Key Data Set (PKDS).

- A _PKISERV_FIPS_LEVEL other than 0 was specified and the CA certificate contains an RSA key smaller than 2048 bits.
- A _PKISERV_FIPS_LEVEL other than 0 was specified and the CA certificate contains a Brainpool ECC key.
- A _PKISERV_FIPS_LEVEL of 1 was specified and the CA certificate contains a NIST ECC key smaller than 192 bits.
- A _PKISERV_FIPS_LEVEL of 2 or 3 was specified and the CA certificate contains a NIST ECC key smaller than 244 bits.
- A _PKISERV_FIPS_LEVEL of 1 was specified and the hash algorithm used on the CA certificate signature is not SHA1 or greater.
- A _PKISERV_FIPS_LEVEL of 2 or 3 was specified and the hash algorithm used on the CA certificate signature is not SHA224 or greater.

System action

PKI Services stops.

System programmer response

Choose a CA certificate that fulfills the requirement according to the FIPS level that is specified by the environment variable _PKISERV_FIPS_LEVEL in the `pkiserv.envars` file. For more information, see section "Requirements for FIPS" on page 12 in Chapter 2.

Routing code

2

Descriptor code

6

IKYP054I PKI SERVICES IS UNABLE TO SET FIPS MODE

Explanation

During PKI Services initialization, it is calling System SSL services to set the FIPS level that is specified in the `pkiserv.envars` file. The call failed.

System action

PKI Services stops.

System programmer response

Look up the error code in *z/OS Cryptographic Services System SSL Programming*. Diagnose the problem indicated by the return code. Restart PKI Services once corrections are made.

Routing code

2

Descriptor code

6

IKYP055I CA CERTIFICATE MAY NOT MEET EST REQUIREMENT

Explanation

During PKI Services initialization, the EnableEST keyword is set to T in `pkiserv.conf`, but the CA certificate does not contain the id-kp-cmcRA extended key usage extension. One of the following requirements must be met for the CA to process EST requests:

1. contains the id-kp-cmcRA extended key usage extension, or
2. contains the domain name in the Subject Alternate Name extension, or the common name in the Subject Distinguished Name, with a value that matches the URI that was used for the request.

   Note: The second condition cannot be checked during initialization time. It will be checked when an EST request comes in. Whether it can be processed or not relies on whether the EST request uses a URI that meets the second requirement.

System action

Initialization continues but PKI Services EST service may not be available if the domain name in the CA certificate does not match that from the request URI.

System programmer response

It is recommended to have an EST CA with the id-kp-cmcRA extended key usage extension. For more information, see Chapter 16, "Using Enrollment over Secure Transport (EST)," on page 341.

Routing code

2

Descriptor code

6

IKYP056I EST CA FILE IS NOT SET UP CORRECTLY

Explanation

During PKI Services initialization, the EnableEST value is set to T in `pkiserv.conf`, and one of the following problems occurs:

1. ESTCAFile is not set, or
2. the file specified by ESTCAFile can not be opened, or
3. the certificate specified by ESTCAFile does not match the CA certificate

System action

Initialization continues but PKI Services EST service may not be available.

System programmer response

Make sure the certificate specified by ESTCAFile is the CA certificate. For more information, see Chapter 16, “Using Enrollment over Secure Transport (EST),” on page 341.

Routing code

2

Descriptor code

6

IKYS001I **Error *nnnn* {attaching | detaching} *OCSF-service-provider-description***

Explanation

PKI Services is attaching or detaching an OCSF or OCEP service provider module. The attach or detach failed. The service provider in error and the error code encountered are displayed.

System action

PKI Services stops.

System programmer response

Look up the error code in either *z/OS Open Cryptographic Services Facility Application Programming* or *z/OS Integrated Security Services Open Cryptographic Enhanced Plug-ins Application Programming*. Diagnose the problem indicated by the return code. Restart PKI Services after corrections are made.

IKYS002I **Error *nnnn* in *OCSF-API-name***

Explanation

PKI Services is calling an OCSF or OCEP API. The invocation has failed. The API name and error code encountered are displayed.

System action

If the error occurs during PKI Services initialization, PKI Services stops. Otherwise, PKI Services continues processing. However, needed cryptographic services are not available.

System programmer response

If you are using ICSF for your CA's private key operations and the failing service is either `CSP_CreateSignatureContext` or `CSSM_SignData`, check that ICSF is functioning and configured properly for PKA operations. For this problem, you also see console message IKYP001E. Follow the instructions for message IKYP001E. For all other errors, look up the error code in either *z/OS Open Cryptographic Services Facility Application Programming* or *z/OS Integrated Security Services Open Cryptographic Enhanced Plug-ins Application Programming*. Diagnose the problem indicated by the return code. Restart PKI Services after corrections are made, if needed.

IKYS003I **Error *nnnn* in getting {subject name | public key} from certificate: *error-code-description***

Explanation

PKI Services is retrieving its CA certificate from the SAF key ring. An error occurred while PKI Services was extracting the subject name or public key from the certificate. The error code encountered is displayed. A description of the error is also displayed, if known. This can indicate a problem with the certificate stored in the SAF key ring or it can be an internal error.

System action

PKI Services stops.

System programmer response

Ensure that the certificate stored in the SAF key ring is correct. If no problems are found, report the error to the IBM support center. For more information, see Chapter 22, “RACF administration for PKI Services,” on page 481 and *z/OS Security Server RACF Security Administrator's Guide*.

IKYS004I **Error 0x*nnnn* in opening key ring *key-ring-name***

Explanation

PKI Services is initializing and is calling System SSL services to open the SAF key ring containing the CA certificate. The open failed. The key ring name and System SSL services error code encountered is displayed.

System action

PKI Services stops.

System programmer response

Look up the error code in *z/OS Cryptographic Services System SSL Programming*. Diagnose the problem indicated by the return code. Restart PKI Services once corrections are made.

IKYS005I Error 0x*nnnn* in closing key ring

Explanation

PKI Services is terminating and is invoking System SSL services to close the SAF key ring containing the CA certificate. The close failed. The System SSL services error code encountered is displayed.

System action

PKI Services continues termination.

System programmer response

Look up the error code in *z/OS Cryptographic Services System SSL Programming*. Diagnose the problem indicated by the return code. Make corrections as indicated. Restart PKI Services if you want.

IKYS006I Cannot delete the signing context

Explanation

PKI Services is attempting to sign a certificate or CRL and is invoking the OCSF API `CSSM_DeleteContext`. The invocation failed.

System action

The certificate or CRL is not created.

System programmer response

Report the error to the IBM support center.

IKYS007I No KeyRing value that is specified under SAF section in pkiserv.conf file

Explanation

PKI Services is reading its configuration file to locate the value that is specified for `KeyRing` in the **SAF** section. The value is missing or has an incorrect syntax.

System action

PKI Services stops.

System programmer response

Correct the value and restart PKI Services if you want. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYS008I Signing key is from unknown crypto service provider

Explanation

PKI Services is retrieving its private key from the SAF key ring. The private key type is not known to PKI Services. This can indicate a problem with the certificate and private key stored in the SAF key ring or it can be an internal error.

System action

PKI Services stops.

System programmer response

Ensure that the certificate and private key stored in the SAF key ring are correct. If no problems are found, report the error to the IBM support center. For more information, see Chapter 22, "RACF administration for PKI Services," on page 481 and *z/OS Security Server RACF Security Administrator's Guide*.

IKYS009I Profile for key ring *key-ring-name* not found

Explanation

PKI Services is reading its configuration file to locate the value that is specified for `KeyRing` in the **SAF** section. The key ring specified is incorrect. No such key ring exists.

System action

PKI Services stops.

System programmer response

Correct the value and restart PKI Services if you want. For more information, see "(Optional) Steps for updating the configuration file" on page 74.

IKYS010I Profile for key ring or default certificate or private key not found

Explanation

PKI Services is attempting to retrieve data from the SAF key ring specified by the `KeyRing` value in the **SAF** section of the `pkiserv.conf` file. The key ring specified does not appear to be set up properly. Possible problems are:

- Key ring is empty.
- CA certificate in the key ring not connected as PERSONAL DEFAULT.
- CA certificate in key ring has no private key.
- User ID assigned to the PKI Services daemon has insufficient authority to read the key ring or private key.

System action

PKI Services stops.

System programmer response

Ensure that the SAF key ring and the certificate stored in it are correct. For more information, see Chapter 4, “Running IKYSETUP to perform RACF administration,” on page 39 and *z/OS Security Server RACF Security Administrator's Guide*.

IKYS011I Error *error-description* in pthread_rwlock_rdlock/wrlock

Explanation

PKI Services is retrieving its CA certificate from the SAF key ring. An internal error occurred while PKI Services was calling the `pthread_rwlock_rdlock` or `pthread_rwlock_wrlock` UNIX function. A description of the error is displayed.

System action

PKI Services stops.

System programmer response

Report the error to the IBM support center.

IKYS012I Error *error-description* in pthread_rwlock_unlock

Explanation

PKI Services is retrieving its CA certificate from the SAF key ring. An internal error occurred while PKI Services was invoking the `pthread_rwlock_unlock` UNIX function. A description of the error is displayed.

System action

PKI Services stops.

System programmer response

Report the error to the IBM support center.

IKYS013I Cannot find the private key associated with the {default | RA} certificate

Explanation

PKI Services is attempting to retrieve data from the SAF key ring specified by the `KeyRing` value in the **SAF** section of the `pkiserv.conf` file. The key ring specified does not appear to be set up properly. The problem is related to either the CA (default) certificate or the RA certificate, as indicated in the message. Possible problems are:

- The certificate is not connected to the ring.
- The certificate is incorrectly connected to the key ring. Both must have USAGE PERSONAL and the CA certificate must be the DEFAULT.
- The certificate has no private key.
- The user ID assigned to the PKI Services daemon has insufficient authority to read the key ring or the private key.

System action

If the problem is with the default certificate, PKI Services stops. If the problem is with the RA certificate, PKI Services continues but the Simple Certificate Enrollment Protocol (SCEP) is disabled.

System programmer response

Ensure that the SAF key ring and the certificate stored in it are correct. For more information, see Chapter 4, “Running IKYSETUP to perform RACF administration,” on page 39 and *z/OS Security Server RACF Security Administrator's Guide*.

IKYS014I Cannot find the {default | RA} certificate with private key associated in key ring

Explanation

PKI Services is attempting to retrieve data from the SAF key ring specified by the `KeyRing` value in the **SAF** section of the `pkiserv.conf` file. The key ring specified does not appear to be set up properly. The problem is related to either the CA (default) certificate or the RA certificate, as indicated in the message. Possible problems are:

- The certificate is not connected to the ring.
- The certificate is incorrectly connected to the key ring. Both must have USAGE PERSONAL and the CA certificate must be the DEFAULT.
- The user ID assigned to the PKI Services daemon has insufficient authority to read the key ring or the private key.

System action

If the problem is with the default certificate, PKI Services stops. If the problem is with the RA certificate, PKI Services continues but the Simple Certificate Enrollment Protocol (SCEP) is disabled.

System programmer response

Ensure that the SAF key ring and the certificate stored in it are correct. For more information, see "Locating your PKI Services certificates and key ring" on page 484 and *z/OS Security Server RACF Security Administrator's Guide*.

IKYS015I **RACF callable service, R_datalib, with function code nnnn returns with SAF return code=*nnnn*, RACF return code=*nnnn*, RACF reason code=*nnnn***

Explanation

PKI Services is attempting to retrieve data from the SAF key ring specified by the `KeyRing` value in the **SAF** section of the `pkiserv.conf` file. The key ring specified does not appear to be set up properly. Possible problems are:

- Key ring is empty.
- CA certificate in the key ring not connected as `PERSONAL DEFAULT`.
- CA certificate in key ring has no private key.
- User ID assigned to the PKI Services daemon has insufficient authority to read the key ring or private key.

System action

PKI Services stops.

System programmer response

Look up the return and reason code that is displayed in *z/OS Security Server RACF Callable Services*. Make corrections as needed. Ensure that the SAF key ring and the certificate stored in it are correct. For more information, see Chapter 22, "RACF administration for PKI Services," on page 481 and *z/OS Security Server RACF Security Administrator's Guide*.

IKYS016I **Error 0x*nnnn* getting the {default | RA | CA} key**

Explanation

PKI Services is initializing and calling System SSL services to get one of its private keys from the key ring. The obtain failed. The problem is related to one of the following, as indicated in the message:

- default - the CA certificate private key
- RA - the RA certificate private key
- CA - a former CA certificate that was re-keyed and rolled over

System action

If either the default or RA key cannot be obtained, PKI Services stops.

If issued for a former CA certificate, PKI Services continues initialization. However, SCEP processing can be disabled if the RA certificate cannot be signature verified.

System programmer response

Look up the error code in *z/OS Cryptographic Services System SSL Programming*. Diagnose the problem indicated by the return code. Restart PKI Services once corrections are made.

IKYS017I **Error 0x*nnnn* exporting the {default | RA} key**

Explanation

PKI Services is initializing and calling System SSL services to export one of its private keys from the key ring. The export failed. The problem is related to either the CA (default) certificate private key or the RA certificate private key, as indicated in the message.

System action

PKI Services stops.

System programmer response

Look up the error code in *z/OS Cryptographic Services System SSL Programming*. Diagnose the problem indicated by the return code. Restart PKI Services once corrections are made.

IKYS018I **Error 0x*nnnn* signing Certificate/CRL**

Explanation

PKI Services is attempting to sign a certificate or CRL. The signing failed.

System action

PKI Services continues processing. However, the needed cryptographic services cannot be available.

System programmer response

If you are using ICSF for your CA's private key operations, check that ICSF is functioning and configured properly for PKA operations. For all other errors, look up the error code in *z/OS Cryptographic Services System SSL Programming*. Diagnose the problem indicated by the return code. Restart PKI Services once corrections are made.

IKYS019I The CA certificate does not have path length constraint capabilities

Explanation

The `EnablePathLenConstraint` keyword was specified in the `pkiserv.conf` configuration file, but the PKI Services CA certificate does not meet the requirements for establishing path length constraint. One or more of the following conditions is true for the CA certificate:

- The key usage extension is present, but the `keyCertSign` bit is not set.
- The basic constraints extension is absent.
- The basic constraints extension is not set correctly in at least one of the following ways:
  - It is not marked critical.
  - The value of `cA` is not true.
  - The value of `pathLenConstraint` is not in the range 0 - 16.

System action:
PKI Services stops.

System programmer response:
Either choose another CA certificate that has the appropriate basic constraints extension, or turn off the `EnablePathLenConstraint` keyword in `pkiserv.conf`. For more information about the `EnablePathLenConstraint` keyword, see "(Optional) Steps for updating the configuration file" on page 74.

IKYS020I The specified PathLength value {none | *value1*} conflicts with the CA path length constraint value *value2*.

Explanation:
During PKI Services initialization, the value that is specified for `PathLength` in the **CertPolicy** section of the `pkiserv.conf` file is validated against the path length constraint value in the basic constraints extension of the PKI Services CA certificate. The `PathLength` value must be positive and smaller than the path length constraint value of the CA certificate. If the `PathLength` value in the message is none, the CA certificate has a path length constraint value but the `PathLength` keyword was not specified in the `pkiserv.conf` file.

System action:
PKI Services stops.

System programmer response:
Specify the value of the `PathLength` keyword in the `pkiserv.conf` file, or correct it to a value smaller than the CA value shown in the message (*value2*), and restart PKI Services. For more information about the `PathLength` keyword, see "(Optional) Steps for updating the configuration file" on page 74.

IKYS021I The value that is specified for the PathLength keyword is not allowed.

Explanation:
The value that is specified for the `PathLength` keyword in the **CertPolicy** section of the `pkiserv.conf` file is not valid. The `PathLength` keyword value must be in the range 0 - 16, and must be less than the `pathLenConstraint` value in the CA certificate Basic Constraint extension, if present.

System action:
PKI Services stops.

System programmer response:
Correct the value that is specified for the `PathLength` keyword in the `pkiserv.conf` file.

IKYS022I This CA is restricted from creating intermediate CA certificates.

Explanation:
The CA certificate in use has a path length constraint value of zero, which prohibits the creation of subordinate or intermediate CA certificates when the `EnablePathLenConstraint` keyword is set to T in the `pkiserv.conf` file.

System action:
None. This message is issued at initialization time to inform you that the CA certificate is restricted from creating CA certificates.

System programmer response
No action is necessary if this configuration is intended. If this configuration is not intended, perform one of the following actions and restart PKI Services:

- Disable path length constraint by removing the `EnablePathLenConstraint` keyword in the `pkiserv.conf` file or by setting its value to F.
- Reconfigure PKI Services to use a CA certificate that does not constrain the path length, or that has the path length greater than zero.

IKYU001I **Unable to open file *file-pathname* for {READ | WRITE}**

Explanation

A user is running a PKI Services utility program. The program is unable the open the input file specified. The name of the file is displayed in the message.

System action

The program ends.

User response

Check that the specified file path name is correct and that the file exists. Also, check the file's permissions to ensure that the user is allowed to process the file. Make changes as necessary and retry the program.

IKYU002I **SAF Service IRRSPX00 Returned SAF RC = *nn* RACF RC = *nn* RACF RSN = *nn* {*diagnostic-information*}**

Explanation

A user is running a PKI Services utility program. The program encountered an error while calling the `R_PKIServ` (IRRSPX00) callable service. The diagnostic information is displayed at the end of the message. For meanings of the diagnostic information, see message IKYI002I.

System action

The program ends.

System programmer response

See message IKYI002I for information. For more information, see Chapter 20, “Using PKI Services utilities,” on page 431.

User response

Correct the problem if applicable. If you cannot correct the problem, contact your web administrator.

RACF administrator response

Determine whether the user should be permitted to perform the task. Make RACF authorization changes if necessary. For authorization information, see “Authorizing users for the PKI Services administration group” on page 481.

Web administrator response

See the web administrator responses listed for message IKYI002I.

IKYU003I **Unknown field name found in SCEP data file, *error-field-name***

Explanation

A user is running the `pkiprereg` PKI Services utility program. The program is reading the SCEP data file and has encountered a field name that it does not recognize. The erroneous field name is displayed.

System action

The program continues. If the user specified the **-l** option, the client is not preregistered.

User response

Correct the data file. Remove any entry that should not be reprocessed and rerun the program if you want. For more information, see “Using the pkiprereg utility” on page 437.

IKYU004I **Required field name {"ClientName" | "Template"} missing from input file at line *nnn***

Explanation

A user is running the `pkiprereg` PKI Services utility program. The program is reading the SCEP data file and encountered a preregistration entry that is missing a required field name. The missing field name is displayed. The line number where the error was found is also displayed.

System action

The program continues. If the user specified the **-l** or **-r** option of the `pkiprereg` utility, the current record is not processed.

User response

Correct the input file. Remove any entry that should not be reprocessed and rerun the program if you want. For more information, see “Using the pkiprereg utility” on page 437.

IKYU005I **Duplicate ClientName=*error-value* entry found in SCEP data file at line *nnn***

Explanation

A user is running the `pkiprereg` PKI Services utility program. The program is reading the SCEP data file and encountered a preregistration entry with a `ClientName` value that is already in use. The value was either used earlier in the file or is already registered in PKI Services. The `ClientName` value and the line number where the error was found are displayed.

System action

The program continues. If the user specified the **-l** option of `pkiprereg` utility, the client is not preregistered.

User response

Correct the input file. Remove any entry that should not be reprocessed and rerun the program if you want. For more information, see "Using the pkiprereg utility" on page 437.

IKYU006I Preregistration record for ClientName=*error-value* not found in PKI Services

Explanation

A user is running the `pkiprereg` PKI Services utility program with the **-r** (remove) option. The program is reading the SCEP data file and encountered a preregistration entry with a `ClientName` value that does not exist in PKI Services. The erroneous field name is displayed.

System action

The program continues. The current record is not processed.

User response

Correct the input file. Remove any entry that should not be reprocessed and rerun the program if you want. For more information, see "Using the pkiprereg utility" on page 437.

IKYU007I Incorrect *field-name=error-value* entry found in SCEP data file

Explanation

A user is running the `pkiprereg` PKI Services utility program. The program is reading the SCEP data file and encountered a field name that has an incorrect value. If the error is length-related, the field name and error value are displayed; otherwise, only the field name is displayed.

System action

The program continues. If the user specified the **-l** option of `pkiprereg` utility, the current entry is not processed.

User response

Correct the input file. Remove any entry that should not be reprocessed and rerun the program if you want. For more information, see "Using the pkiprereg utility" on page 437.

IKYU008I Template nickname *template* not found in certificate templates file

Explanation

A user is running the `pkiprereg` PKI Services utility program. The program is reading the SCEP data file and encountered a value pair of *template=nickname* where the nickname is not found in the certificate templates file.

System action

The program continues. If the user specified the **-l** option of `pkiprereg` utility, the client is not preregistered.

User response

Correct the input file. Remove any entry that should not be reprocessed and rerun the program if you want. For more information, see "Using the pkiprereg utility" on page 437.

IKYU009I Out of memory

Explanation

A user is running a PKI Services utility program. The program is unable to allocate more memory.

System action

The program ends.

System programmer response

Increase the amount of REGION available to the user or adjust the Language Environment® runtime memory settings. (For instructions, see *z/OS Language Environment Programming Guide*.)

User response

Increase the amount of memory available to the program if possible. Otherwise, report the error to your system programmer.

IKYU010I **Internal error occurred in function *function-name*: *diagnostic-information***

Explanation

A user is running a PKI Services utility program. The program has encountered an internal error. The function in error is displayed. Additional diagnostic information can also be displayed.

System action

The program ends.

User response

Report the error to the IBM support center.

IKYU011I **System function *function-name* detected error -- *error-string***

Explanation

A user is running a PKI Services utility program. The program received an error calling a system service. The service name and error message are displayed.

System action

The program ends.

User response

See documentation related to the service that failed. Make any necessary corrections. Then, rerun the program.

IKYU012I **Unable to open message catalog *message-catalog-filename* -- *error-string***

Explanation

A user is running a PKI Services utility program. The program is attempting to open an external message catalog using the default location for the message catalog as set by the NLSPATH environment variable.

System action

The program continues using default messages.

System programmer response

If the user requires an external message catalog, correct the indicated error. There are several reasons that could cause this error, such as file or directory permissions not allowing read access.

For information about updating the NLSPATH environment variable, see *z/OS UNIX System Services Programming Tools*. If default messages are acceptable, no action is necessary.

User response

If you require an external message catalog, correct the indicated error. Then, rerun the program. For system errors, report the problem to your system programmer. For more information, see Chapter 20, “Using PKI Services utilities,” on page 431.

IKYU013I **Incorrect command syntax. The *error-value* parameter is {unknown | missing | incorrectly specified} Usage: *command-usage***

Explanation

A user is running a PKI Services utility program. A command parameter was entered incorrectly. The correct command usage is displayed.

System action

The program ends.

User response

Execute the command with the correct syntax. For more information, see Chapter 20, “Using PKI Services utilities,” on page 431.

IKYU014I **pkiprereg complete. *nnn* preregistration records processed. *mmm* successful. *000* errors found**

Explanation

A user is running the `pkiprereg` PKI Services utility program. The program completed and is reporting the results. The value of *nnn* is the total number of preregistration records (entry groups) found in the SCEP data file. The value of *mmm* is the number of preregistration records that were successfully processed. The value of *000* is the number of errors found. Because multiple errors can occur for each preregistration record, *mmm* plus *000* can exceed *nnn*.

System action

The program ends.

User response

If no errors were found, no action is required. If errors were found, correct the errors in the SCEP data file. (You can remove the entries that were successful.) Rerun the program with the corrected SCEP data file.

For more information, see Chapter 20, "Using PKI Services utilities," on page 431.

IKYU015I Template with nickname *template* cannot be used for preregistration

Explanation

A user is running the `pkiprereg` PKI Services utility program. The program is reading the certificates template file and found the certificate template with the specified nickname.

- If you are using CGI web pages and pkiserv.tmpl, the template either has no **<PREREGISTER>** section or the **<PREREGISTER>** section is not properly terminated.
- If you are using JSP web pages and PKIServTemplate.xml, the `certreq_template` tag for this nickname did not include a valid `preregrules` element.

System action

The program continues. If the **-l** option was specified, the client is not preregistered.

System programmer response

Correct the certificate template. Edit the SCEP data file to remove any entry that should not be reprocessed and rerun the program if you want. For more information, see "Using the pkiprereg utility" on page 437.

IKYU016I pkiprereg complete. *nnn* passwords generated

Explanation

A user is running the `pkiprereg` PKI Services utility program in **generate** mode (password generate mode). The program completes and reports the results. *nnn* is the total number of passwords that are generated.

System action

The program ends.

User response

No action is required.

IKYU017I *program-name* terminated abnormally

Explanation

A user is running a PKI Services utility program. The program cannot complete due to a user or system error. The name of the utility program is displayed in the message.

System action

The program ends.

User response

Check for previously issued messages and their associated actions. Make changes as necessary and retry the program.

IKYU018I Conversion from VSAM to DB2 failed at record *record-key* in VSAM file *vsam-file-name*

Explanation

A user is running the PKI Services utility program `vsam2db2` to convert the existing object store and ICL VSAM files to Db2 tables. An error occurred when processing the record from the VSAM file indicated in the message. This message should be preceded by one indicating whether the source of the problem is VSAM or Db2.

System action

The `vsam2db2` utility program ends.

User response

Fix the error, drop the Db2 tables, and rerun the `vsam2db2` utility. For more information, see "Using the vsam2db2 utility" on page 448.

If you cannot fix the error based on the information in the message, report it to the IBM support center.

IKYU019I Conversion to target storage failed at record *record-number* in [ObjectStore | ICL]

Explanation

The PKI Services utility program `vsamconv` or `db2conv` encountered a failure while converting the record indicated in the message. This message is preceded by another message that indicates the source of the problem.

System action

The utility program ends. The target VSAM data sets or Db2 tables may contain records up to the point where the failure occurred.

User response

Check for the preceding failure message and perform the associated actions, then clear the target data sets or Db2 tables and rerun the utility program. For more information, see “Using the vsamconv utility” on page 450 or “Using the db2conv utility” on page 432. If you cannot repair the failure based on the information in the message, report the error to the IBM support center.

Chapter 27. File directory structure

This topic contains information about the location of files in:

- z/OS product libraries
- File system directory `/usr/lpp/pkiserv/` and its subdirectories.

Product libraries

SMP/E installs PKI Services into the following product libraries:

- SAMPLIB/ASAMPLIB
  - IKYALLOC
  - IKYCDB2
  - IKYCDBV1
  - IKYCVSAM
  - IKYCVSV1
  - IKYDDDEF
  - IKYISMKD
  - IKYMKDIR
  - IKYRVSAM
  - IKYRVSV1
  - IKYSBIND
  - IKYSETUP
  - IKYSGRNT
  - IKYVBKUP
  - IKYVREST
- PROCLIB/APROCLIB
  - IKYSPROC with alias PKISERVD
- SIEALNKE/AIEALNKE
  - IKYPKID - The PKI Services daemon
  - IKYPRTM - The Resource Termination Manager for the daemon
- CBRDBRM/ACBRDBRM
  - IKYPDBRM
  - IKYPDBR1

File system directory and subdirectories

Unless you change the default, SMP/E installs PKI Services into the file system directory `/usr/lpp/pkiserv`. Table 106 on page 604 describes the directory structure and contents:

Table 106. Files contained in subdirectories

Subdirectory	Contains...
`ActiveX`	The unsigned PKI Services ActiveX program installation files and the source files for use when creating signed ActiveX program installation files. For more information, see Appendix C, "Using the PKI Services web application with Internet Explorer on Windows systems," on page 691. `ActiveX` contains the following subdirectories: • `signsrc` contains the unsigned ActiveX control files for Microsoft Internet Explorer (files named PKICEnroll.*). It also contains the file PKIActiveX.lic, which is used for all versions of Windows. Use these files to build signed ActiveX program installation files. • `PKICEnroll` contains the unsigned ActiveX control installation files for Microsoft Internet Explorer.
`bin`	Utilities: • `createcrls` - Program to create certificate revocation lists (CRLs). (For more information, see "Using the createcrls utility" on page 431.) • `iclview` - Utility for viewing issued certificate list (certificate database). (For more information, see "Using the iclview utility" on page 433.) • `pkiprereg` - Utility for creating preregistration records in batch for Simple Certificate Enrollment Protocol (SCEP) and Enrollment over Secure Transport (EST) clients, see "Using the pkiprereg utility" on page 437.) • `pkitp_install` - Program to register the PKI Services Trust Policy plug-in with OCSF. (For more information, see "Configuring and getting started with PKITP" on page 511.) • `pkitp_ivp` - Program to verify that the PKI Services Trust Policy plug-in installed successfully. (For more information, see "Configuring and getting started with PKITP" on page 511.) • `postcerts` - Program to post issued certificates to an LDAP directory. (For more information, see "Using the postcerts utility" on page 441.) • `TemplateTool` - Utility that works with certificate template files to perform functions you need if you implement the web application using JavaServer pages (JSPs). (For more information, see "Using the TemplateTool utility" on page 442.) • `vosview` - Utility for viewing the object store (request database). (For more information, see "Using the vosview utility" on page 444.) • `vsam2db2` - Program to copy data from the issued certificate list (ICL) and object store VSAM data sets into Db2 tables. (For more information, see "Using the vsam2db2 utility" on page 448.)
`include`	C header files: • `pkitp.h` - C language header file for writing application programs that use the PKI Trust Policy Plug-in. (For more information, see "Files for PKITP" on page 511.)

Table 106. Files contained in subdirectories (continued)

Subdirectory	Contains...
`lib`	Libraries and message catalogs: • `pkitp.so` - OCSF Trust Policy plug-in for PKI Services. (For more information, see "Files for PKITP" on page 511. • `*.dll` - Dynamic link libraries (DLLs) that the PKI Services daemon uses. • `nls/msg/En_US.IBM-1047/*.cat` - The PKI Services message catalogs. (These message catalogs are also symbolically linked in the `/usr/lpp/pkiserv/lib/nls/msg/C` directory and in the `/usr/lib/nls/msg/En_US.IBM-1047` and `/usr/lib/nls/msg/C` directories.) • `*.jar` - Java archives. • `librpkisJNI.so` - JNI shared object.
`lib64`	64 bit library: • `librpkisJNI64.so` - 64-bit JNI shared object.
`pkijsp`	The JavaServer page (JSP) enterprise archive (EAR) file. (For information about the EAR file, see "(Optional) Modifying the JSP files and the EAR file" on page 258).
`PKIServ`	CGIs that make up the PKIServ web application. (For information about CGIs, see "Relationship between CGIs and the pkiserv.tmpl file" on page 217 and Table 43 on page 233.) `PKIServ` contains the following subdirectories: • `public-cgi` - Public (non-SSL) directory • `ssi-cgi-bin` - SSL-protected – `auth` - SSL with user ID and password protection. Work runs under client's ID. – `surrogateauth` - SSL with user ID and password protection. Work runs under surrogate ID (PKISERV). • `clientauth-cgi-bin` - SSL with client certificate protection. Work runs under surrogate ID (PKISERV). – `auth` - SSL with client certificate protection. Work runs under administrator's ID.

Table 106. Files contained in subdirectories (continued)

Subdirectory	Contains...
`samples`	Various sample files, including: • `expiringmsg.form` - The email message sent to a user as notification about a certificate that expires • IBM HTTP Server - Powered by Apache configuration files: `httpd.conf` - HTTP Server configuration file `vhost80` - Virtual host configuration file for non-SSL requests. `vhost443` - Virtual host configuration file for SSL requests with server authentication. `vhost1443` - Virtual host configuration file for SSL requests with client authentication. • `Makefile.pkiexit` - The makefile for the PKI Services exit. (For more information, see Chapter 17, "Customizing with installation exit routines," on page 349.) • `Makefile.pkitpsamp` - The makefile for pkitpsamp.c, which is a sample application to call the PKI Trust Policy plug-in. (For more information, see "Files for PKITP" on page 511.) • `pendingmsg.form` - The email message sent to an administrator listing pending requests. • `pendingmsg2.form` - The email message sent to an administrator listing pending requests that are modified. • `pkiexit.c` - The sample PKI Services exit, which PKI Services provides. (For more information, see Chapter 17, "Customizing with installation exit routines," on page 349.) • `pkiserv.envars` - The PKI Services environment variables file. (For more information, see "Optionally updating PKI Services environment variables" on page 70 and "The pkiserv.envars environment variables file" on page 617.) • `pkiserv.tmpl` - The PKI Services certificate templates file used with the REXX CGI execs. (For more information, see Chapter 11, "Customizing the end-user web application if you use REXX CGI execs," on page 135.) • `pkiserv.conf` - The PKI Services configuration file. (For more information, see "(Optional) Steps for updating the configuration file" on page 74 and Chapter 28, "The pkiserv.conf configuration file," on page 607.) • `PKIServ.xsd` - The XML schema for the PKI Services XML template, `pkitmpl.xml`. (For more information, see Chapter 13, "Implementing the web application using JavaServer pages," on page 237.) • `pkitmpl.xml` - The PKI Services XML template file used with JavaServer pages (JSPs). (For more information, see Chapter 13, "Implementing the web application using JavaServer pages," on page 237.) • `pkitpsamp.c` - Sample application to call the PKI Trust Policy plug-in. (For more information, see "Files for PKITP" on page 511 and "Building the sample application to invoke the certificate validation service" on page 515.) • `readymsg.form` - The email message sent to a user as notification a certificate is ready for retrieval • `recoverymsg.form` - The email message sent to a user who has requested that PKI Services recover a certificate for which PKI Services generated the keys • `rejectmsg.form` - The email message sent to a user as notification a request for a certificate has been rejected • `renewcertmsg.form` - The email message sent to a user as notification that PKI Services has automatically renewed an expiring certificate

Chapter 28. The pkiserv.conf configuration file

This topic includes a code sample of the `pkiserv.conf` configuration file.

The `pkiserv.conf` file is the configuration file for the PKI Services daemon. By default, you can find this file in the `/usr/lpp/pkiserv/samples/` directory. For more information about the sections of the `pkiserv.conf` configuration file and the parameters, see "(Optional) Steps for updating the configuration file" on page 74 and Table 21 on page 74.

The following listing might not be identical to the code sample shipped with the product. For the most current sample, see the `pkiserv.conf` file in the source directory `/usr/lpp/pkiserv/samples/`.

```
# Licensed Materials - Property of IBM
# 5650-ZOS
# Copyright IBM Corp. 2001, 2019
# Status = HKY77C0

[OIDs]
#
# Supported Distinguished Name OIDs
#
C=2.5.4.6
O=2.5.4.10
OU=2.5.4.11
CN=2.5.4.3
L=2.5.4.7
ST=2.5.4.8
TITLE=2.5.4.12
POSTALCODE=2.5.4.17
STREET=2.5.4.9
MAIL=0.9.2342.19200300.100.1.3
EMAIL=1.2.840.113549.1.9.1
SERIALNUMBER=2.5.4.5
UNSTRUCTUREDNAME=1.2.840.113549.1.9.2
UNSTRUCTUREDADDRESS=1.2.840.113549.1.9.8
DNQUALIFIER=2.5.4.46
DC=0.9.2342.19200300.100.1.25
UID=0.9.2342.19200300.100.1.1
BUSINESSCATEGORY=2.5.4.15
JURISDICTIONCOUNTRY=1.3.6.1.4.1.311.60.2.1.3
JURISDICTIONSTATEPROV=1.3.6.1.4.1.311.60.2.1.2
JURISDICTIONLOCALITY=1.3.6.1.4.1.311.60.2.1.1

#
# Signature Algorithm OIDs
#
sha-1WithRSAEncryption=1.2.840.113549.1.1.5
sha-256WithRSAEncryption=1.2.840.113549.1.1.11
sha-384WithRSAEncryption=1.2.840.113549.1.1.12
sha-512WithRSAEncryption=1.2.840.113549.1.1.13
sha-224WithRSAEncryption=1.2.840.113549.1.1.14
md-5WithRSAEncryption=1.2.840.113549.1.1.4
md-2WithRSAEncryption=1.2.840.113549.1.1.2
id-dsa-with-sha1=1.2.840.10040.4.3
id-dsa-with-sha224=2.16.840.1.101.3.4.3.1
id-dsa-with-sha256=2.16.840.1.101.3.4.3.2
ecdsa-with-sha1=1.2.840.10045.4.1
ecdsa-with-sha224=1.2.840.10045.4.3.1
ecdsa-with-sha256=1.2.840.10045.4.3.2
ecdsa-with-sha384=1.2.840.10045.4.3.3
ecdsa-with-sha512=1.2.840.10045.4.3.4
#
# RSASSA-PSS Signature algorithm OID and related hash algorithm OIDs
#
rsassa-pss=1.2.840.113549.1.1.10
sha-256Hash=2.16.840.1.101.3.4.2.1
sha-384Hash=2.16.840.1.101.3.4.2.2
sha-512Hash=2.16.840.1.101.3.4.2.3

# If your organization will be using CertificatePolicies extensions
# on certificates that are created by this CA, the following
# entry assigns a symbolic name to a registered OID that identifies
# your organization's certificate usage policy.  This symbolic name
# is used later in the [CertPolicy] section of this configuration
# file to specify the certificate policy information.
#
MyPolicy=1.2.3.4

[ObjectStore]
# Database implementation, either VSAM or DB2. Default is VSAM.
# Specify
# DBType=VSAM
# or
# DBType=DB2
#
# Database version, either 0 or 1.  Default is 0.
# Specify
# DBVersion=0
# or
# DBVersion=1

# If DBType is DB2, configure the following additional keywords:
#     DBPackage                 DBSubsystem       DBWaitTime
# These keywords will be ignored if DBType is set to VSAM.
#
# If DBType is VSAM, configure the following additional keywords:
#     ObjectDSN                 ICLDSN
#     ObjectStatusDSN           ICLStatusDSN
#     ObjectRequestorDSN        ICLRequestorDSN
#     ObjectSCEPTidDSN          ICLSCEPTidDSN
#     ObjectTidDSN
# These keywords will be ignored if DBType is set to DB2.
#
# Regardless of the setting for DBType, verify the setting of
# the following keywords in this ObjectStore section:
#     SharedPLEX                RemoveCompletedReqs
#     RemoveInactiveReqs        RemoveExpiredCerts
#     RemoveExpiredCertsAndKeys

# Is the database implementation, whether it be VSAM datasets
# or DB2, shared in a sysplex with other instances of PKI
# Services?  True (T) or False (F).
# Note: The SharedPLEX keyword below replaces the SharedVSAM
#       keyword in PKI Services V1R13.  This keyword has the
#       same meaning as the SharedVSAM keyword has in prior
#       version of PKI Services.
#
SharedPLEX=F

# Name of the DB2 Package this instance of PKI will be using in
# in the DB2 sub system specified by the DBSubsystem keyword. If
# DBSubsystem is missing or not specified, this keyword will be
# ignored.
#
# DBPackage=MasterCA

# Name of the DB2 Sub system. If DBPackage is missing or not
# specified, this keyword will be ignored.
#
# DBSubsystem=DSN9

# How long in days (d),  hours (h) or minutes (m) should PKI wait for
# DB2 to be available before it shuts down?
# The default value is 0m which indicates PKI does not wait. It stops
# when DB2 is not available. The maximum wait time is 1 day.
# This keyword will be ignored if DBType is not DB2.
# DBWaitTime=30m

# Data set name of the VSAM request (object store) base CLUSTER
#
ObjectDSN='pkisrvd.vsam.ost'

# Data set name of the VSAM object store PATH for the transaction ID
# (TID) alternate index.
#
ObjectTidDSN='pkisrvd.vsam.ost.path'

# Data set name of the VSAM object store PATH for the status alternate
# index
#
ObjectStatusDSN='pkisrvd.vsam.ost.status'

# Data set name of the VSAM object store PATH for the requestor
# alternate index
#
ObjectRequestorDSN='pkisrvd.vsam.ost.requestr'

# Data set name of the VSAM object store PATH for the SCEP Transaction
# ID alternate index
#
#ObjectSCEPTidDSN='pkisrvd.vsam.ost.sceptid'

# Data set name of the VSAM issued certificate list (ICL) base CLUSTER
#
ICLDSN='pkisrvd.vsam.icl'

# Data set name of the VSAM ICL PATH for the status alternate index
#
ICLStatusDSN='pkisrvd.vsam.icl.status'

# Data set name of the VSAM ICL PATH for the requestor alternate index
#
ICLRequestorDSN='pkisrvd.vsam.icl.requestr'

# Data set name of the VSAM ICL PATH for the SCEP Transaction ID
# alternate index
```

```
#
#ICLSCEPTidDSN='pkisrvd.vsam.icl.sceptid'

# How many days (d) or weeks (w) should completed requests remain in
# the object store before being removed?
# Specify 0d to indicate completed requests should not be removed
#
RemoveCompletedReqs=1w

# How many days (d) or weeks (w) should inactive requests remain in the
# object store before being removed?
# Specify 0d to indicate inactive requests should not be removed
#
RemoveInactiveReqs=4w

# How many days (d) or weeks (w) should expired certificates remain in
# the ICL before being removed?
# Specify 0d to indicate expired certificates should not be removed
#
#RemoveExpiredCerts=26w

# How many days (d) or weeks (w) should expired certificates and Keys
# remain in the ICL and TKDS? Specify 0d to indicate expired
# certificates and keys should not be removed
#
#RemoveExpiredCertsAndKeys=520w

[CertPolicy]
# What signature algorithm should be used to sign certificates that are
# created?  The name of the signature algorithm must match one of the
# Signature Algorithm OIDs listed in the [OIDs] section of this
# configuration file.
#
# If an rsassa-pss algorithm is desired, then a hash algorithm is also
# required. The supported hash algorithm from the OID section (sha-256Hash,
# sha-384Hash, or sha-512Hash) must be specified after the rsassa-pss
# algorithm, separated by a comma. For example:
#     SigAlg1=rsassa-pss,sha-256Hash
#
SigAlg1=sha-256WithRSAEncryption

# How often should the certificate creation thread scan the database
# for approved certificate requests?
#
CreateInterval=3m

# How many days or weeks prior to the expiration of a certificate should
# the expiration warning be sent. If not specified, expiration warning
# will not be sent.
#
ExpireWarningTime=4w

# How often should certificate revocation lists (CRL) be created?
#
TimeBetweenCRLs=1d

# How long is a certificate revocation list (CRL) valid?
#
CRLDuration=2d

# Specify the number of certificates that each CRL distribution point
# will represent. The default is 0 which indicates distribution point
# CRLs should not be created.
#
CRLDistSize=500

# Specify the constant portion of the CRL distribution point relative
# distinguished name. The distribution point number is appended to this
# value to form the common name. The default value is "CRL".
#
CRLDistName=CRL

# Should an Authority Revocation List(ARL) Distribution Point be
# created? 'F' (default) indicates an ARL DP will not be created.
# 'T' indicates an ARL DP will be created if the CRLDistSize is
# greater than zero.
#
ARLDist=F

# Full path of the directory where CRL distrubution point files are to
# be stored for http protocol URI CRL distribution points.
# Defaults to "/var/pkiserv/"
# Ignored if no http protocol CRLDistURIn are defined
#
CRLDistDirPath=/var/pkiserv/

# Values for the CRL distribution point extension URI fields for the
# protocols(ldap, http) you choose. This is repeatable. The first one
# always starts with CRLDistURI1,  followed by CRLDistURI2, 3, ...n,
# if necessary.  Uncomment and update the desired directive to enable
# URI CRL distribution point that you need. If more than one URI field
# is needed, remember to increase the field number sequentially by the
# order of one, e.g. CRLDistURI2, CRLDistURI3...
# For ldap protocol, you may specify the LDAP server indicated in the LDAP
# section below, e.g.,
#
#CRLDistURI1=LdapServer1

# or specify a skeleton URL which contains the protocol type, the domain
# name and the port, if needed, e.g.,
#
#CRLDistURI1=ldap://myotherldapserver.mycompany.com:389/

# For http protocol, specify the complete URL minus the file name of the
# distribution point CRL file, e.g.,
#
#CRLDistURI1=http://www.mycompany.com/PKIServ/cacerts/

# Enable large (>32KB) CRL posting support which will store CRLs in
# a local directory prior to being posting to LDAP.
# T - True, CRLs are stored in a local file system directory before
#           being posted to LDAP
# F - False, CRLs are stored in Object Store posting object record(s)
#            before being posted to LDAP. Warning, a CRL (distribution
#            point or master) larger than 32KB will fail to be created
#            and not be posted to LDAP. (This is the Default)
#
#EnableLargeCRLPosting=F

# Full path of the local directory where CRLs are saved prior to
# posting to LDAP.
# Defaults to /var/pkiserv/.
# This keyword is ignored if large CRL posting is not enabled.
#
#LargeCRLPostPath=/var/pkiserv/

# Should this CA create CRLs that contain a critical Issuing
# Distribution Point extension?
# T = True, CRLs will be created with a critical Issuing Distribution
#           Point extension. (This is the default value if not
#           specified.)
# F = False, CRLs will be created with no Issuing Distribution Point
#            extension.
#
#CRLIDPExt=T

# What type of OCSP request is desired?
#  'none'  - No OCSP responder support (This is the default)
#  or
#  'basic' - The OCSP responder is enabled, but will not verify the
#            optional request signature.
#
OCSPType=none

# Enable the Simple Certificate Enrollment Protocol (SCEP)
# T = True, SCEP is enabled
# F = False, SCEP is disabled (default if not specified)
#
EnableSCEP=F

# Enable the Certificate Management Protocol (CMP)
# T = True, CMP is enabled
# F = False, CMP is disabled (default if not specified)
#
EnableCMP=F

# Enable the Enrollment over Secure Transport Protocol (EST)
# T = True, EST is enabled
# F = False, EST is disabled (default if not specified)
#
EnableEST=F

# Specify the full pathname containing the EST CA file in DER format,
# required if EnableEST=T
#
#ESTCAFile=/var/pkiserv/estcacert.der

# Specify the template nickname corresponds to the EST template
# in pkiserv.tmpl or pkitmpl.xml for EST preregistration and
# certificate fulfillment. Maximum length is 8.
# Ignored if the value is greater than 8 characters or if
# EST is not enabled.
#
#ESTTemplate=2YESTP

# Should the CA restrict certificate requests to a validity period
# that does not exceed the CA certificate life time?
# T = True, requests with a validity period that exceeds the CA's
#           will fail.
# F = False, requests are not constrained to the CA's validity
#            period(this is the default value if not specified)
#
#CertValidityConstraint=F

#
```

```
# Should certificate path length constraints be enabled/enforced
# by this CA?
# T = True, The CA certificate will be examined at initialization
#           to verify it meets path length constraint requirements
#           and enables the setting of the pathLenConstraint
#           field in the Basic Constraint extension of intermediate
#           CA certificates created by this CA.
# F = False, Certificate path length constraints will not be
#            enforced in the CA certificate used by this CA, and
#            intermediate CA certificates created by this CA will
#            not include a pathLenConstraint field in the Basic
#            Constraint extension. (this is the default value if
#            not specified)
#
#EnablePathLenConstraint=F

#
# Specify the certification path length constraint value to be
# included in the Basic Constraints extension of intermediate
# CA certificates created by this CA.
# - The EnablePathLenConstraint keyword must be set to T, otherwise
#   the PathLength keyword will be ignored.
# - The valid value range for this keyword is 0 to 16, however the
#   value specified must be less than the pathLenConstraint value
#   in the PKI CA certificate if it is present.
#
#PathLength=1

# CertificatePolicies certificate extension information, indicating the
# policy under which the certificate has been issued and the purposes
# for which the certificate may be used. This extension contains a
# sequence of one or more policy information terms, each term comprised
# of an OID and an optional qualifier.

# Should the CA require that the CertificatePolicies extension be
# included on all certificates that are created?
# T = True, the CertificatePolicies extension will be added to all
#             certificates, and will include all PolicyName<n> entries
#             specified in this file.  Any policies that are
#             specified in the CertPolicies input parameter or listed
#             in the CONSTANT section of the template used to
#             generate the certificate are ignored.
# F = False,  the CertificatePolicies extension will only be added
#             to certificates when a certificate policy is specified
#             in the CertPolicies input parameter or in the
#             CONSTANT section of the template when a certificate is
#             requested. (This is the default value)
#
PolicyRequired=F

# Should the CertificatePolicies certificate extension be made a
# critical extension?
# T = True, the extension will be marked Critical
# F = False, the extension will not be marked Critical (This is the
#            default)
#
PolicyCritical=F

# List of CertificatePolicies extensions identifiers that may be added
# to certificates created by this CA.
# The policy name is the symbolic name for a certificate policy OID
# and must match the name of a policy that is listed in the [OIDs]
# section of this configuration file.
#
PolicyName1=MyPolicy

# Should the CertificatePolicies certificate extension include
# any optional qualifiers?  Qualifiers may be Certification
# Practice Statement (CPS) Pointer and User Notice.  User Notice
# may have two optional fields: Notice Reference and Explicit Text.
# To include these optional qualifiers for certificates created
# using the certificate policy <n>, uncomment the appropriate
# entries below and tailor them to suit your purpose.  Note that
# for the CA to conform with current standards, Notice Reference
# should not be used.
#
# CPS<n> = Specifies the URI for the CPS associated with PolicyName<n>.
# Policy<n>Org = Names the organization that has prepared the User
#                      Notice Reference information associated
#                      with PolicyName<n>.
# Policy<n>Notice<m> = Identifies the number of a textual
#                      statement, prepared by Policy<n>Org,
#                      for the User Notice Reference associated
#                      with PolicyName<n>.  More than one
#                      textual statement may apply.
# UserNoticeText<n> =  Specifies the User Notice Explicit Text
#                      information associated with PolicyName<n>.
#                      For the CA to conform with current standards,
#                      this textual statement must not exceed 200
#                      characters.
#
#Policy1Org=MyOrganization
#Policy1Notice1=3
```

```
#Policy1Notice2=17
UserNoticeText1=This is some very lawyerly statement for the relying party to read and make decisions based
on.
CPS1=http://www.mycompany.com/cps.html

# Length of certificate suspension grace period in day or weeks (d,w).
# Certificates which remained suspended for longer than this period are
# automatically revoked.
# The default value is 0d which indicates the grace period is unlimited.
#
MaxSuspendDuration=120d

# Specify the email address of an administrator who will receive an
# email notification when a certificate request state becomes pending
# approval. Repeat for each administrator to receive pending approval
# notifications. The first one always starts AdminNotifyNew1, followed
# by AdminNotifyNew2, 3, ...n, if necessary. The field number increases
# sequentially by the order of one. Uncomment and update the desired
# email address to enable the notification.
#
#AdminNotifyNew1=adminA@abc.com

# Specify the email address of an administrator who will receive an
# email notification containing a list of requests that are pending
# approval when the maintenance processing is run. Repeat for each
# administrator to receive an email reminder. The first one always
# starts AdminNotifyReminder1, followed by AdminNotifyReminder2, 3,
# ...n, if necessary. The field number increases sequentially by the
# order of one. Uncomment and update the desired email address to
# enable the notification.
#
#AdminNotifyReminder1=adminA@abc.com

# Enable granular authority control for the administrative functions on
# different templates. If enabled, appropriate RACF(or equivalent
# product) protection profiles must be set up accordingly.
# T = True, granular authority control is enabled
# F = False, granular authority control is disabled (default if not
#            specified)
#
#AdminGranularControl=F

# Should a console message be issued when the CRL processing finishes?
# none - No console message will be issued
#        (this is the default value if not specified)
# file - A console message will be issued after the CRLs are available
#        in the file system
# This keyword will be ignored unless large CRL posting support is
# enabled (EnableLargeCRLPosting=T) or at least one http protocol CRL
# distribution point URI is defined.
#
#CRLWTONotification=none

#
# Which certificates should be included in the PKCS#12 package when
# PKI Services has generated the public/private key pair?
# I = The issuing CA certificate is included with the end-entity
#     and private key. (This is the default behavior if not specified)
# E = Only the end-entity certificate and private key will be included
#     in the PKCS#12 package.
# C = The complete chain up to the Root that is connected to the CA
#     keyring is included with the end-entity certificate and private
#     key in the PKCS#12 package.
#
#PKCS12Content=I

[General]
InitialThreadCount=10

# Timeout value for the exit program. Default is 30 seconds (30s).
ExitTimeout=30s

# full pathname or data set name containing the 'your certificate is
# ready to be retrieved' message form. Defaults to no message issued
ReadyMessageForm=/etc/pkiserv/readymsg.form

# full pathname or data set name containing the 'your certificate
# request has been rejected' message form. Defaults to no message issued
RejectMessageForm=/etc/pkiserv/rejectmsg.form

# full pathname or data set name containing the 'your certificate is
# about to expire' message form. Defaults to no message issued
ExpiringMessageForm=/etc/pkiserv/expiringmsg.form

# full pathname or data set name containing the request(s) pending for
# approval message form. Defaults to no notification sent.
AdminNotifyForm=/etc/pkiserv/pendingmsg.form

# full pathname or data set name containing the request(s) approved
# with modifications message form. Defaults to no notification sent.
AdminNotifyModForm=/etc/pkiserv/pendingmsg2.form

# full pathname or data set name containing the renewed certificate
# message form for automatic certificate renewal.
# If absent, automatic certificate renewal is disabled.
RenewCertForm=/etc/pkiserv/renewcertmsg.form

# full pathname or data set name containing information on
# the list of certificates that match the criteria specified
# to recover key generated certificates.
# If absent, recovery query results will not be sent.
RecoverForm=/etc/pkiserv/recoverymsg.form

# Time of day to run the PKI maintenance task in 24 hour time format
# (HH:MM). The valid range is 00:00-23:59. The default value is 00:00
# (midnight).
#MaintRunTime=01:00

# Days of the week to run the PKI maintenance task in 0-6 format. The
# value specified is a list of numbers between 0 and 6. 0 represents
# Sunday and 6 represents Saturday.  No spaces or any other characters
# are permitted. Order of the digits is not important. Repeat digits
# are not allowed.
# The default value is everyday of the week: 0123456
#MaintRunDays=0123456

# Should the PKI maintenance task run when the PKI daemon is started?
# True (T) or False (F). Default value is True.
#RunMaintAtStart=T


[SAF]
KeyRing=PKISRVD/CAring
#TokenName=PKISRVD.PKIToken

# The Label name for the PKI RA certificate connected to the Key ring
#   specified in the KeyRing value above
#
RALabel=Local PKI RA

# Should the CA generate secure keys in the Token Data Set (TKDS)
# when it has key generation capability?
# Valid SecureKey values are:
# T - True indicates secure keys are generated in the TKDS
# F - False (or absence of this keyword) indicates clear keys
#     will be generated in the TKDS. Note: Installation
#     configuration policy may override the ability to create
#     clear keys causing clear key requests to create secure
#     keys.
# If TokenName is not specified, the SecureKey keyword
# will be ignored.
# SecureKey=T

[LDAP]
NumServers=1
PostInterval=5m
Server1=myldapserver.mycompany.com:389
AuthName1=CN=root
AuthPwd1=root
#
# Should the CA post certificates and CRLs to the LDAP server with the
# binary attribute?
# T = True, post certificates and CRLs with the binary attribute
# F = False, post certificates and CRLs without the binary attribute
#            (this is the default value if not specified)
# Note: If NumServers is greater than one, you need one value for
# each corresponding server, eg. UseBinaryAttr1 is for Server1.
# If the corresponding UseBinaryAttrn is missing, it defaults to F.
# UseBinaryAttr1=F
CreateOUValue= Created by PKI Services
RetryMissingSuffix=T
# Name of the LDAPBIND Class profile containing the bind information
# for LDAP server 1. This key is optional. Used in place of keys
# Server1, AuthName1, and AuthPwd1.
#BindProfile1=LOCALPKI.BINDINFO.LDAP1
```

Chapter 29. Environment variables

This topic describes the environment variables that PKI Services uses and their possible values. It also includes a code sample of the environment variables file, `pkiserv.envars`. (See "The pkiserv.envars environment variables file" on page 617.) For information about the PKISERVD procedure, which specifies the path name of the environment variables file, see "PKISERVD sample procedure to start PKI Services daemon" on page 682.

Environment variables in the environment variables file

The environment variables contained in `pkiserv.envars` and their values are:

_PKISERV_CA_DOMAIN

Specifies the CA domain. The first eight characters must be unique. The first eight characters of the CA domain name are limited to the following character set: alphanumeric characters (`a-z`, `A-Z`, `0-9`) and the hyphen (`-`). In addition, the first character must not be a number or hyphen.

Example:

```
_PKISERV_CA_DOMAIN=WebAppCA
```

_PKISERV_CONFIG_PATH

Specifies the path name for the directory containing the configuration file, `pkiserv.conf`, and the certificate template file, `pkiserv.tmpl` for this CA domain. The default value (if you do not set the environment variable) is `/etc/pkiserv`.

Guideline: Copy both of these files from the install directory, `/usr/lpp/pkiserv/samples`, before making any changes.

Note: Because the PKISERV CGIs run in an IBM HTTP Server address space, if the `pkiserv.tmpl` file is not in its default location of `/etc/pkiserv/pkiserv.tmpl`, you need to add the `_PKISERV_CONFIG_PATH` variable to the IBM HTTP Server environment variable file.

- If you are using IBM HTTP Server - Powered by Apache, the environment variables are added to the configuration files using the `SetEnv` directive.

_PKISERV_EXIT

Specifies the full path name for the installation-provided PKI exit program that the PKI Services daemon invokes to perform autorenew preprocessing or postprocessing. (This exit is a UNIX-executable program or shell script.) If you do not define this variable or if it contains a null value, the PKI autorenew exit processing is disabled.

Note: The _PKISERV_EXIT environment variable is also used by the PKI Services CGI scripts to specify an exit program to be used by the web application. The PKI Services CGI scripts run in an IBM HTTP Server address space, so you must specify the _PKISERV_EXIT environment variable in the IBM HTTP Server environment variables file.

- If you are using IBM HTTP Server, the environment variables are added to the configuration files using the `SetEnv` Directive.

_PKISERV_FIPS_LEVEL

Specifies the FIPS compliance level to be used by PKI Services. Four compliance levels are permitted.

0

No FIPS compliance is enforced at this level. This is the default if _PKISERV_FIPS_LEVEL is not specified.

1
FIPS 140-2 compliance is enforced at this level.

2
SP800-131A with exception is enforced at this level.

3
SP800-131A without exception is enforced at this level.

Guideline: Before modifying the value of _PKISERV_FIPS_LEVEL environment variable, ensure that all pending certificate requests have been completed. Some existing requests may not comply with the new FIPS level.

Example:

```
_PKISERV_FIPS_LEVEL=2
```

_PKISERV_MSG_LOGGING
Values include:

STDOUT_LOGGING
Indicates writing all messages (verbose, diagnostic, informational, warning, error, and severe) to STDOUT and additionally writing the error and severe messages to STDERR. This is the default if the environment variable is not set.

STDERR_LOGGING
Indicates writing verbose, diagnostic, informational, and warning messages to STDOUT and writing error and severe messages to STDERR.

_PKISERV_MSG_LEVEL
Specifies the subcomponent and message level to log. Messages for a particular subcomponent are logged only if the message level is greater than or equal to the specified level for that subcomponent. You can use an asterisk (*) to indicate all subcomponents. The subcomponent list consists of a subcomponent name and a message level separated by a period (.).

For example, the following sets the message level for all subcomponents to log warning messages or higher. (This is the default setting.)

Example:

```
_PKISERV_MSG_LEVEL=*.W
```

You can specify multiple subcomponents by separating entries with a comma (,). For example, the following indicates that all subcomponents are set to message level **W** (warning) and that the PKID subcomponent is set to message level **D** (diagnostic).

Example:

```
 _PKISERV_MSG_LEVEL=*.W,PKID.D
```

The subcomponents are:

Subcomponent	Meaning
*	The wildcard character (represents all subcomponents)
CORE	The core functions of PKI Services that are not specific to the other subcomponents
DB	Activity related to the object store or issued certificate list repositories
LDAP	LDAP posting operations
PKID	The PKI Services daemon address setup and infrastructure
POLICY	Certificate creation and revocation policy processing

Subcomponent	Meaning
SAF	SAF key ring, OCEP, and R_datalib calls
TPOLICY	Trust policy plug-in processing

The message levels are listed hierarchically:

Debug level	Meaning
S	This indicates logging only severe messages.
E	This indicates logging severe and error messages.
W	This indicates logging severe, error, and warning messages. This is the *default* message level for all subcomponents if you do not set the environment variable.
I	This indicates logging severe, error, warning, and informational messages.
D	This indicates logging severe, error, warning, informational, and diagnostic messages.
V	This indicates logging *all* messages, including verbose diagnostic messages. This is very verbose. **Guideline:** Do not use **V** level unless IBM support personnel instruct you to do so.

_PKISERV_VARDIR
Specifies the path name for a directory in which PKI Services will write persistent data. The maximum length of the path name is 256 characters, including the trailing /. The default value (if you do not set the environment variable) is /var/pkiserv.

_PKISERV_ENABLE_JSP

Specifies whether you use the JSP interface and XML templates for PKI Services web pages, or the REXX CGI execs and text templates. Set to TRUE to use the JSP interface. The default (if you do not set the environment variable) is to use the REXX CGI exec interface.

The pkiserv.envars environment variables file

The following code sample is for the pkiserv.envars environment variables file. (For information about updating the environment variables file, see "Optionally updating PKI Services environment variables" on page 70.) The following listing might not be identical to the code sample shipped with the product. For the most current sample, see the pkiserv.envars file in the source directory /usr/lpp/pkiserv/samples/.

```
#----------------------------------------------------------------------#
#                                                                      #
# PKI Services sample environment variable file                        #
#                                                                      #
# Licensed Materials - Property of IBM                                 #
# 5650-zOS                                                             #
# Copyright IBM Corp.  2001, 2019                                      #
# Status = HKY77C0                                                     #
#                                                                      #
#----------------------------------------------------------------------#
#
# Language and Path configurations
#
#
LANG=En_US.IBM-1047
PATH=/bin
```

```
LIBPATH=/usr/lpp/pkiserv/lib:/usr/lib
NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lpp/pkiserv/lib/nls/msg/%L/%N

#
# When running as a CA Domain, set the CA Domain name by assigning
# desired value to the _PKISERV_CA_DOMAIN variable.
# Note: The first eight characters must be unique.
#
# example: _PKISERV_CA_DOMAIN=WebAppCA

#
# Configuration File location and Message configuration Options
#
_PKISERV_CONFIG_PATH=/etc/pkiserv
_PKISERV_MSG_LOGGING=stdout_logging
_PKISERV_MSG_LEVEL=*.w

#
# Set up a directory for PKI Services to write persistent data. The
# maximum length is 256 characters including the trailing /.
# The default is /var/pkiserv/.
#
# example: _PKISERV_VARDIR=/var/pkiserv/

#
# Set up an exit program for autorenew. The maximum length is 256
# characters including the program name.
#
# example: _PKISERV_EXIT=/mydir/renewexit

#
# Enable the JSP Webpages and XML Template.
#
#_PKISERV_ENABLE_JSP=TRUE

# Set FIPS operational level. Possible values:
#     0    FIPS is not active
#     1    FIPS 104-2 is active
#     2    SP800-131A with exception is active
#     3    SP800-131A without exception is active
#
#_PKISERV_FIPS_LEVEL=0
```

Chapter 30. The IKYSETUP REXX exec

IKYSETUP is a REXX exec that issues RACF commands to perform RACF administration. This topic describes the actions IKYSETUP performs and provides a code sample of IKYSETUP.

Actions IKYSETUP performs by issuing RACF commands

In broad terms, the actions that IKYSETUP performs are as follows:

- Sets up the PKI Services daemon user ID
- Sets up the access control to protect PKI Services
  - Protects end-user functions
  - Protects administrative functions
- Defines one or more CA domains with associated administrative domains
- Creates the CA and RA certificates, their private keys, and key ring
- Creates the IBM HTTP Server certificate, private key, and key ring
- Enables surrogate operation for the IBM HTTP Server
- Allows PKI Services to generate key pairs for certificate requests

Setting up the PKI Services daemon user ID

Create the daemon user ID (by default, PKISRVD) using the RACF ADDUSER TSO command. Give it an OMVS segment because it needs access to z/OS UNIX. If you implement the object store and issued certificate list (ICL) using VSAM data sets, this user ID also needs update access to the VSAM data sets identified in the **ObjectStore** section of the `pkiserv.conf` file. If necessary, use the RACF ADDSD and PERMIT TSO commands to give this user ID UPDATE access to the VSAM data sets. If you implement the object store and ICL using Db2 tables, this user ID also needs access to the Resource Recovery Services Access Facility (RRSAF). If necessary, use the RACF RDEFINE and PERMIT commands to define the profile for RRSAF in the DSNR class and give this user ID READ access.

Guideline: Define the daemon user ID with the NOPASSWORD attribute.

To associate this user ID to the PKI Services started procedure, use the following RACF TSO commands:

```
RDEFINE  STARTED PKISERVD.* STDATA(USER(PKISRVD))
SETROPTS CLASSACT(STARTED) RACLIST(STARTED)
SETROPTS RACLIST(STARTED) REFRESH
```

Setting up access control to protect PKI Services

This task can be divided into two steps:

1. Protecting end-user functions
2. Protecting administrative functions.

Protecting end-user functions

You must first determine who your end-users are and how they are using their certificates. In general there are two categories of end-users:

- Internal clients, such as employees who have SAF user IDs on the host system and who might be using their certificates to access resources on the host
- External clients, who have no access to the host system.

When PKI Services is called, the unit of work has some identity (user ID) associated with it. For external customers, a surrogate user ID is necessary.

Guideline: Although under certain circumstances it might be beneficial for internal clients to access PKI Services under their own identities, your implementation is simpler if you use surrogate user IDs for internal clients also.

Use the RACF ADDUSER command to create the surrogate user ID (PKISERV). Give it an OMVS segment because it needs access to z/OS UNIX. **Guideline:** Define the surrogate user ID with the PROTECTED and RESTRICTED attributes.

The `R_PKIServ` SAF callable service is protected by FACILITY class resources of the form IRR.RPKISERV.*function*[.*ca_domain*], where function is one of the following and *ca_domain* specifies an optional CA domain name. (Specify *ca_domain* when your installation has established multiple PKI Services CAs.)

The `R_PKIServ` functions are:

EXPORT
Retrieves (exports) a previously requested certificate, or retrieves (exports) the PKI Services registration authority (RA) certificate or the certificate authority (CA) certificate.

GENCERT
Generates an auto-approved certificate.

GENRENEW
Generates an auto-approved renewal certificate. (The request submitted is automatically approved.)

QRECOVER
Lists certificates whose key pairs were generated by PKI Services under a requestor's email address and passphrase.

REQCERT
Requests a certificate that an administrator must approve before it is created.

REQRENEW
Requests certificate renewal. The administrator needs to approve the request before the certificate is renewed.

RESPOND
Invokes the PKI OCSP responder.

REVOKE
Revokes a certificate that was previously issued.

SCEPREQ
Generates a certificate request using Simple Certificate Enrollment Protocol (SCEP).

VERIFY
Confirms that a given user certificate was issued by this certificate authority and, if so, returns the certificate fields.

Create these resources and give the PKISERV user ID either READ or CONTROL access to them. CONTROL bypasses subsequent resource checks.

Additional FACILITY class resources of the form IRR.DIGTCERT.*function* protect the actual certificate generation and retrieval functions. If subsequent resource checks are not being bypassed, define these resources and their access.

There are two ways to handle certificate approval:

- An administrator can review certificate requests
- Requests can be auto-approved without administrator action (this should probably be reserved for internal clients only).

If you plan to have an administrator approve certificate requests before issuing certificates, PKISERV needs the following access:

Table 107. Access required if you plan to have an administrator approve certificate requests

Resource	Access
IRR.DIGTCERT.REQCERT	READ
IRR.DIGTCERT.REQRENEW	READ

If your clients can request certificates that are auto-approved without action by an administrator, PKISERV needs the following access:

Table 108. Access required if you plan to use auto-approval

Resource	Access
IRR.DIGTCERT.ADD	UPDATE
IRR.DIGTCERT.GENCERT	CONTROL
IRR.DIGTCERT.GENRENEW	READ

Finally, because the web server is switching identities to PKISERV, you must give it surrogate permission. This is done by creating another resource in the SURROGAT class (BPX.SRV.PKISERV) and giving the web server daemon user ID READ access to it.

Protecting administrative functions

PKI Services administrators must have SAF user IDs on the host system. When PKI Services is called for administrative functions, the unit of work is tagged with the identity of the authenticated administrator.

At a minimum, all PKI Services administrators require READ or UPDATE access to the profile IRR.RPKISERV.PKIADMIN[.*ca_domain*] in the FACILITY class. Table 109 on page 621 shows how the level of access to this profile controls authorization to general administrative functions.

Table 109. FACILITY class access needed for administrative functions

Resource	Access	Purpose
IRR.RPKISERV.PKIADMIN[.*ca_domain*]	READ	For list and query operations
	UPDATE	To act on certificate requests, preregistration requests, and issued certificates

In addition, you can use profiles in the PKISERV class to restrict PKI Services administrator access to specific operations. For information, see "Using the PKISERV class to control access to administrative functions" on page 499. By default this additional capability is not enabled. The `AdminGranularControl` keyword in the `pkiserv.conf` configuration file controls whether it is enabled.

Example: To grant user ID ADMID authority to administer the PKI Services CUSTOMER domain, and to grant that same user the ability to query information about PKI Services certificates issued using the "1-Year PKI SSL Browser Certificate" template, issue the following RACF TSO commands:

```
RDEFINE  FACILITY (IRR.RPKISERV.PKIADMIN.CUSTOMER) UACC(NONE)
PERMIT   IRR.RPKISERV.PKIADMIN.CUSTOMER CLASS(FACILITY) ACCESS(UPDATE) ID(ADMID)
RDEFINE  PKISERV CUSTOMER.QUERYCERTS.1YBSSL UACC(NONE)
PERMIT     CUSTOMER.QUERYCERTS.1YBSSL ACCESS(READ) CLASS(PKISERV) ID(ADMID)
SETROPTS RACLIST (FACILITY) REFRESH
SETROPTS CLASSACT(PKISERV)  RACLIST(PKISERV)
SETROPTS RACLIST (PKISERV)  REFRESH
```

Establishing your CA and RA certificates

To create and sign digital certificates for others, you need to establish a CA certificate, and optional RA certificate, and their associated private keys using the RACDCERT command.

Steps for establishing your CA and RA certificates

Perform the following steps to create your CA certificate, RA certificate, and their associated keys, back up the keys, connect them to a key ring and authorize PKI Services to use them.

Before you begin

Determine the CA or RA's distinguished name and where it will be located (under CERTAUTH for the CA and under the PKI Services daemon user ID for the RA). Typically, CAs and RAs have distinguished names in the following form:

`OU=`*`your-CA-or-RA's-friendly-name`*`.O=`*`your-organization`*`.C=`*`your-two-letter-country-abbreviation`*

Procedure

1. Create your CA certificate, and optional RA certificate, and their associated private keys using the RACDCERT GENCERT command. If you create an optional RA certificate, it must be signed by the CA certificate.

   a. This example creates a 20-year CERTAUTH certificate with a distinguished name of `OU=Human Resources Certificate Authority.O=Your Company, Inc.C=US`.

   ```
   RACDCERT CERTAUTH GENCERT SUBJECTSDN(
      OU('Human Resources Certificate Authority')
      O('Your Company, Inc')  C('US')) WITHLABEL('Local PKI CA') HIGHTRUST
      NOTAFTER(DATE(2026/05/06))
      SIZE(1024) KEYUSAGE(HANDSHAKE)
      SIGNWITH(CERTAUTH LABEL('Local Root CA'))
   ```

   b. This example creates a 20-year RA certificate signed by the CA certificate created in Example "1.a" on page 622.

   ```
   RACDCERT GENCERT ID(PKISERVD) SUBJECTSDN(
      CN('Registration Authority')
      OU('Human Resources Certificate Authority')
      O('Your Company, Inc')  C('US')) WITHLABEL('Local PKI RA')
      NOTAFTER(DATE(2026/05/06))
      SIZE(1024) KEYUSAGE(HANDSHAKE)
      SIGNWITH(CERTAUTH LABEL('Local PKI CA'))
   ```

2. Back up your CA certificate, RA certificate (if created), and their associated private keys to password-protected data sets using the RACDCERT EXPORT command.

   ```
   RACDCERT CERTAUTH EXPORT(LABEL('Local PKI CA'))
      DSN('PKISRVD.PRIVATE.KEY.P12BIN')
      FORMAT(PKCS12DER) PASSWORD('your-passphrase')

   RACDCERT ID(PKISRVD) EXPORT(LABEL('Local PKI RA'))
      DSN('PKISRVD.PRIVATE.RAKEY.P12BIN')
      FORMAT(PKCS12DER) PASSWORD('your-passphrase')
   ```

3. When using ICSF for private key protection in Public Key Data Set (PKDS) and signing, use the RACDCERT ADD command. For this step to be successful, ICSF CCA cryptographic coprocessor must be operational and configured with Public Key Data Set (PKDS). (For additional information about ICSF, see *z/OS Cryptographic Services ICSF Administrator's Guide*.)

   ```
   RACDCERT CERTAUTH ADD('PKISRVD.PRIVATE.KEY.P12BIN') PASSWORD('your-passphrase')
   PKDS
   ```

```
RACDCERT CERTAUTH ADD('PKISRVD.PRIVATE.RAKEY.P12BIN') PASSWORD('your-passphrase')
PKDS
```

4. Create a key ring for the PKI Services daemon and add the CA certificate and RA certificate (if created) to it so that PKI Services can use the certificates. The example creates a key ring that is called `CAring` for user ID PKISRVD and connects the CA and RA certificates to it.

   Important: Make sure that your CA certificate is marked with the TRUST or HIGHTRUST attribute in RACF. (Otherwise, PKI Services cannot use the certificate.) Check this by issuing the RACDCERT LIST command and execute the RACDCERT ALTER command to change it if needed.

```
RACDCERT ADDRING(CAring) ID(PKISRVD)

RACDCERT ID(PKISRVD) CONNECT(CERTAUTH LABEL('Local PKI CA') RING(CAring)
   USAGE(PERSONAL) DEFAULT)

RACDCERT ID(PKISRVD) CONNECT(ID(PKISRVD) LABEL('Local PKI RA') RING(CAring)
   USAGE(PERSONAL))
```

5. Authorize the PKI Services daemon to use RACF certificates and act as the CA by using the following FACILITY class resource. RACLIST the FACILITY class if it is not already RACLISTed.

```
SETROPTS RACLIST(FACILITY)
RDEFINE FACILITY IRR.DIGTCERT.GENCERT
PERMIT IRR.DIGTCERT.GENCERT  CLASS(FACILITY) ID(PKISRVD) ACCESS(CONTROL)
SETROPTS RACLIST(FACILITY) REFRESH
```

   The daemon user ID (PKISRVD) also needs access to the CA certificate through its key ring by using the RDATALIB or FACILITY class resources as shown in the following example.

   If the RDATALIB class is used, RACLIST the class if it is not already RACLISTed:

```
SETROPTS RACLIST(RDATALIB)
RDEFINE RDATALIB PKISRVD.CAring.LST
PERMIT PKISRVD.CAring.LST CLASS(RDATALIB) ID(PKISRVD) ACCESS(CONTROL)
SETROPTS RACLIST(RDATALIB) REFRESH
```

   If the FACILITY class is used:

```
RDEFINE FACILITY IRR.DIGTCERT.LISTRING
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(PKISRVD) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

Configuring the IBM HTTP Server - Powered by Apache for SSL mode

The PKISERV application requires the IBM HTTP Server - Powered by Apache to operate in three modes. The modes are:

- Normal
- SSL without client authentication
- SSL with client authentication.

For SSL, your server needs to obtain a digital certificate. You can:

- Purchase one from an external source
- Create one using RACF

Note: If your server is already operating in SSL mode, you can skip the following section, "Using RACF to obtain a certificate for the web server" on page 623.

Using RACF to obtain a certificate for the web server

The IBM HTTP Server supports using either gskkyman key databases (`.kdb` files) or RACF (SAF) key rings for the server's certificate store. You are expected to use SAF key rings if setting up their web server for the first time.

Note: If you have already set up your web server using gskkyman, you can continue to use it.

Use RACDCERT to generate the server certificate signed by the new Certificate Authority.

Example:

```
RACDCERT GENCERT ID(WEBSRV) SIGNWITH(CERTAUTH LABEL('Local PKI CA'))
   WITHLABEL('SSL Cert') SUBJECTSDN(CN('www.YourCompany.com') O('Your Company Inc')
   L('Millbrook') SP('New York') C('US'))
```

The web server needs a key ring containing its new certificate and any trusted CA certificate. The RACDCERT command with operands ADDRING and CONNECT also sets this up. For example, the RACDCERT commands to create a key ring that is called `SSLring` for user ID WEBSRV and to connect the web server and CA certificates to it are:

Example:

```
RACDCERT ADDRING(SSLring) ID(websrv)
RACDCERT ID(websrv) CONNECT(CERTAUTH LABEL('Local PKI CA')) RING(SSLring)
   USAGE(PERSONAL) DEFAULT)
RACDCERT ID(websrv) CONNECT(ID(websrv) LABEL('SSL Cert') RING(SSLring)
   USAGE(PERSONAL) DEFAULT)
```

Export the CA certificate to an MVS data set. Then OPUT it to a file system file so that it can be made available to your clients.

Example:

```
RACDCERT EXPORT(LABEL(''Local PKI CA'))
   CERTAUTH DSN('pkisrvd.webroot.derbin') FORMAT(CERTDER)
```

Enabling the IBM HTTP Server - Powered by Apache for surrogate operation

Your server must be able to act as a surrogate for clients requesting certificates. To enable this, create:

- Profile BPX.SERVER in the FACILITY class
- Profile BPX.SRV.PKISERV in the SURROGAT class.

Give the IBM HTTP Server - Powered by Apache daemon user ID READ access to both of these profiles.

Allowing PKI Services to generate key pairs for certificate requests

You can choose to allow PKI Services to generate key pairs (public and private key) for certificate requests. The PKI Services daemon does this using the PKCS #11 API provided by ICSF. Set up profiles in the CRYPTOZ class to allow the PKI Services daemon to use the PKCS #11 API:

- Activate the CRYPTOZ class.
- Define the profile SO.*daemon_id*.* in the CRYPTOZ class.
- Give the daemon user ID UPDATE access to the profile
- Define the profile USER.*daemon_id*.* in the CRYPTOZ class.
- Give the daemon user ID CONTROL access to the profile.

IKYSETUP sample

IKYSETUP contains the commands to perform the RACF administrator tasks of adding groups and user IDs, setting up access control, creating CA, RA, and SSL certificates, and setting up daemon security. The following listing might not be identical to the code sample shipped with the product. For the most current sample, see SYS1.SAMPLIB member IKYSETUP.

```
/* REXX */
/*********************************************************************/
/*                                                                   */
/* DESCRIPTIVE NAME:  PKI Services RACF setup CLIST                  */
/*                                                                  */
/* Licensed Materials - Property of IBM                             */
/* 5650-ZOS                                                         */
/* Copyright IBM Corp. 2001, 2019                                   */
/* Status = HKY77C0                                                 */
/*                                                                  */
/*01* EXTERNAL CLASSIFICATION: OTHER                                */
/*01* END OF EXTERNAL CLASSIFICATION:                               */
/*                                                                  */
/* FUNCTION:                                                        */
/*                                                                  */
/*   This CLIST will issue the RACF TSO commands necessary to set up*/
/*   security for PKI Services. It must be run from TSO by a user ID*/
/*   that is RACF SPECIAL.                                          */
/*                                                                  */
/* USAGE:                                                           */
/*                                                                  */
/*   1) Read accompanying PKI Services post installation            */
/*      instructions.                                               */
/*   2) Perform necessary prerequisite product installation for     */
/*      the webserver (websphere), LDAP, etc.                       */
/*   3) Make note of any predetermined values such as the LDAP      */
/*      suffix, webserver fully qualified domain name, and the      */
/*      settings contained in the pkiserv.conf file.                */
/*   4) Copy the CLIST to a data set where you can edit it.         */
/*   5) Examine the entire CLIST, in particular, the configurable   */
/*      section.                                                    */
/*   6) Modify the values in the configurable section as needed for */
/*      your installation.                                          */
/*   7) Run the CLIST. Syntax:                                      */
/*                                                                  */
/*  EX 'data-set-name(IKYSETUP)' 'RUN(YES | NO | PROMPT)'           */
/*                                                                  */
/*  where:  YES - indicates to run CLIST as is                      */
/*          NO - indicates to display the commands only             */
/*          PROMPT - indicates to prompt the user prior             */
/*           to invoking each command                               */
/*                                                                  */
/* DISCLAIMER:                                                      */
/*                                                                  */
/*   This CLIST is not intended to cover every possible customer    */
/*   scenario. Modification of the actual commands to be issued     */
/*   may be required                                                */
/*                                                                  */
/********************************************************************/

trace value('O')

/*--------------------------------------------------------------*/
/* configurable section                                         */
/*--------------------------------------------------------------*/

/*--------------------------------*/
/* Part 1 - Things you must change */
/*--------------------------------*/


/**************************************************************/
/* This exec will create the certificate, private key,  and     */
/* keyring needed for your certificate authority.               */
/*                                                              */
/* You must update the distinguished name of your certificate   */
/* authority defined below. The suffix of this DN must match    */
/* the suffix set up for your LDAP directory (suffix value from */
/* your slapd.conf file).                                       */
/*                                                              */
/* Typically, Certificate Authorities have distinguished names  */
/* in the following form:                                       */
/*                                                              */
/*  OU=<your-CA's-friendly-name>,O=<your-organization>,         */
/*        C=<your-2-letter-country-abbreviation>                */
/*                                                              */
/* e.g., OU=Human Resources Certificate Authority.O=IBM,C=US    */
/*                                                              */
/* If you already have your CA certificate and private key set  */
/* up in RACF, set ca_dn="" and update the ca_label variable to */
/* equal your CA certificate's label. Note, it must reside      */
/* under CERTAUTH                                               */
/*                                                              */
/* If you are running with Multiple-CAs:                        */
/*   You could run IKYSETUP once for each separate CA you       */
/*   want to operate, changing ca_domain every time. The        */
/*   ca_domain value will help qualify the other variables thus */
/*   reducing the amount of work the RACF administrator needs   */
/*   to perform. Otherwise, set to NULL.                        */
/*                                                              */
/****************************************************************/
ca_domain = ""                                          /* @L4A*/

if LENGTH(ca_domain) > 8 then                           /* @L4A*/
   ca_domain_trunc = LEFT(ca_domain,8)                  /* @L4A*/
else                                                    /* @L4A*/
   ca_domain_trunc = ca_domain                          /* @L4A*/


OrgUnit = STRIP(ca_domain "Human Resources Certificate Authority")
                                                        /* @L4A*/
ca_dn= "OU('"||OrgUnit||"')",
       "O('Your Company')",
       "C('Your Country 2 Letter Abbreviation')"        /* @L4C*/

ca_label = STRIP(ca_domain "Local PKI CA")  /* Label for CA
                                             certificate with the
                                             CA Domain name
                                             prepended      @L4A*/

/****************************************************************/
/* ra_label:                                                    */
/*   A "must change" variable - default: "Local PKI RA"         */
/* ra_dn:                                                       */
/*   A "must change" variable. If you don't wish to have PKI    */
/*   Services operate with a separate RA certificate, set       */
/*   ra_dn=""                                                   */
/****************************************************************/

ra_label = STRIP(ca_domain "Local PKI RA") /*Label for
                                             RA Certificate @01C*/

if (ra_label = "") then      /* If no RA Label ...          @L4A*/
   ra_dn=""                                              /*@L4A*/
else                                                     /*@L4A*/
   ra_dn=,                                               /*@L4A*/
     "CN('Registration Authority')",
      ca_dn

/****************************************************************/
/* This exec will create the certificate, private key,  and     */
/* keyring needed for your webserver. (Required for SSL.)       */
/*                                                              */
/* You should set up the Domain name in the Subject Alternative */
/* Name(SAN) extension of your webserver certificate to match   */
/* its fully qualified domain name. Otherwise you must set up   */
/* the Common Name(CN) in the Subject Distinguished Name(SDN)   */
/* to match.                                                    */
/* e.g., ALTNAME(DOMAIN('www.YourCompany.com')) or              */
/*       CN=www.YourCompany.com,O=Your Company,C=US             */
/*                                                              */
/* You must specify the webserver’s domain name with the        */
/* web_host variable if:                                        */
/* - the webserver certificate is to be created in this exec,   */
/*   or                                                         */
/* - set up a CA certificate for Enrollment over Secure         */
/*   Transport(EST) processing (see for details in Part 3)      */
/* web_host is used to set up web_altdomain (Domain in SAN) or  */
/* web_cn (Common Name in SDN) or both in creating the          */
/* webserver certificate. It is also used to create the EST CA  */
/* certificate.                                                 */
/*                                                              */
/* If needed, web_ip can be used to set up web_altip (IP in SAN)*/
/* instead of web_host.                                         */
/*                                                              */
/* web_ip is used to set up web_altip (Domain in SAN) or        */
/* web_cn (Common Name in SDN) or both in creating the          */
/* webserver certificate. It is also used to create the EST CA  */
/* certificate.                                                 */
/*                                                              */
/* If you already have your webserver configured for SSL, set   */
/* web_dn="".                                                   */
/****************************************************************/
/*web_host = "" */
web_host = www.YourCompany.com

web_ip = ""
/*web_ip = 9.56.214.7 */
```

```
web_altdomain = ""
if web_host ^=  "" then
  web_altdomain=”DOMAIN('"web_host"')"

web_altip = ""
if web_ip ^=  "" then
 web_altip = "IP("web_ip")"

web_cn = ""
if web_host ^=  "" then
 web_cn = "CN('"web_host"')"
else if web_ip <> "" then
 web_cn = "CN('"web_ip"')"

web_dn=,
 web_cn,
 "O('Your Company')",
 "L('Your City')",
 "SP('Your Full State or Province Name')",
 "C('Your Country 2 Letter Abbreviation')"

/*****************************************************************/
/* If you want the CA to support Enrollment Over Secure         */
/* Transport (EST) features, the Domain value in the Subject    */
/* Alternative Name extension of the CA certificate must match  */
/* the host name in the URI that will be presented to the EST   */
/* CGI.                                                          */
/* Set ca_host value below to construct the ALTNAME DOMAIN      */
/* extension for the CA certificate.                             */
/*                                                               */
/* If needed, set ca_ip value below to construct the ALTNAME IP */
/* extension for the CA certificate.                             */
/*                                                               */
/* For example, if EST clients will be contacting the following */
/* URI for the cacerts function to obtain the CA certificate    */
/* chain from your CA:                                           */
/*                                                               */
/*    https://www.YourCompany.com/.well-known/est/cacerts       */
/*                                                               */
/* The value that you should specify for "ca_host" is:          */
/*                                                               */
/*    ca_host = "www.YourCompany.com"                           */
/*                                                               */
/* The value of ca_host (or ca_ip) should be the same as the    */
/* Domain (or IP) value in the Subject Alternative Name         */
/* extension of the webserver certificate, if it is present;    */
/* otherwise ca_host (or ca_ip) needs to match the webserver’s  */
/* Common Name in the Subject Distinguished Name, although this */
/* is not recommended.                                           */
/*                                                               */
/*****************************************************************/
ca_altdomain = ""
ca_altip = ""
if web_host = "" & web_ip = "" then do
  call logsay "Unable to set up EST CA SAN extension without ",
              "web_host and web_ip value"
end

else do
  if web_host ^=  "" then do
    ca_host = web_host
    ca_altdomain = ”DOMAIN('"ca_host"')"
  end
  if web_ip ^=  "" then do
    ca_ip = web_ip
    ca_altip = "IP("ca_ip")"
  end
end

/*****************************************************************/
/* The sample web server protection directives supplied by PKI  */
/* use SSLring for the web server's SAF key ring. If you change */
/* the value below, you will need to modify   the "KeyFile"     */
/* directive in the samples/vhost443.conf and                   */
/* samples/vhost1443.conf files when configuring the web server.*/
/*                                                          @LGC */
/* If you already have your webserver configured for SSL and    */
/* are using a SAF key ring (vs a gskkyman keyfile), then set   */
/* web_ring equal to your webserver's SAF key ring name. If you */
/* are using a gskkyman keyfile, then set web_ring="". Note,    */
/* you will have to manually add the PKI CA's certificate to    */
/* the webserver's keyfile/keyring and make the webserver's     */
/* root CA available in an HFS file for download.          @L9C */
/****************************************************************/
web_ring = "SSLring"              /* SAF keyring for web server */

/****************************************************************/
/* You must provide UID and GID values for the user IDs and     */
/* groups being created below                                   */
/****************************************************************/
daemon="PKISRVD"                  /* user ID for PKI daemon     */
daemon_uid="554"                  /* uid for PKI daemon         */
surrog="PKISERV"                  /* user ID for the surrogate  */
surrog_uid="555"                  /* uid for the surrogate id   */

/****************************************************************/
/* pkigroup members are authorized to administer PKI Services   */
/* certificates and certificate requests.  If you know the user */
/* IDs that should be connected to this group, update the       */
/* pkigroup_mem stem variable. If not, you can always connect   */
/* users later.                                                 */
/*                                                              */
/* If you do not wish to have this exec create this group,      */
/* set the group name to ""                                     */
/*                                                              */
/****************************************************************/
pkigroup="PKIGRP"             /* PKI Services Admin group name  */
pki_gid="655"                 /* PKI Services Admin group id    */
pkigroup_mem.0=0       /* Number of pkigroup members to connect */
pkigroup_mem.1=""


/*-----------------------------------*/
/* Part 2 - Questions you must answer */
/*-----------------------------------*/

/****************************************************************/
/* Question 1 -  Restrict the surrogate user ID?                */
/*                                                              */
/* The surrogate user ID is the identity assigned to client     */
/* processes when requesting certificate services. The          */
/* RESTRICTED attribute can be assigned to this ID to limit the */
/* resources available to this user should the user ID be       */
/* hijacked by an unfriendly client (hacker). We recommend      */
/* that you run the surrogate this way. However, this probably  */
/* will cause additional setup work. If you want the RESTRICTED */
/* attribute assigned now, set restrict_surrog=1. Note, you     */
/* can always do this at some later time.                       */
/****************************************************************/
restrict_surrog=0

/****************************************************************/
/* Question 2 -  Need PKI Services to generate the key pair     */
/*               for its certificates?                          */
/*                                                              */
/* You have the option of choosing PKI Services to generate the */
/* key pair for the certificates using PKCS#11 APIs. In order   */
/* for the daemon to generate the key pair, you need to specify */
/* the TokenName keyword in pkiserv.conf and the CRYPTOZ class  */
/* must be activated, the profiles must be defined and the      */
/* daemon needs to have appropriate access.                     */
/*                                                              */
/* If you don't need the key generation capability, leave       */
/* key_gen = 0, otherwise set key_gen = 1.                 @L9A */
/*                                                              */
/* If you set key_gen = 1, you must also specify a CSFSERV      */
/* profile via csfserv_profile set later in this file.     @D8A */
/*                                                              */
/****************************************************************/
key_gen = 0                                              /*@L9A*/

/****************************************************************/
/* If you set key_gen=1 above, this exec will activate the      */
/* CRYPTOZ class, create profiles in this class and permit      */
/* the PKI Services daemon to access them. You may also have    */
/* a RACF group for authorized PKCS11 token users. The daemon   */
/* ID would need to be added to this group.                     */
/*                                                              */
/* Set the following variables as needed:                       */
/*                                                              */
/* cryptoz_profile_so - Profile to be created in the CRYPTOZ    */
/*     class to control the security officer (SO) role          */
/* cryptoz_profile_user - Profile to be created in the CRYPTOZ  */
/*     class to control the user role                        */
/* cryptoz_grp - Group name for authorized PKCS11 token users   */
/*                                                       @L9A */
/**************************************************************/
cryptoz_profile_so = 'SO.'||daemon||'.*'                 /*@L9A*/
cryptoz_profile_user = 'USER.'||daemon||'.*'             /*@L9A*/
cryptoz_grp = ''                                         /*@L9A*/

/**************************************************************/
/* Question 3 - What key type to use for the CA certificate?    */
/*                                                            */
/* There are several possible choices for generation and        */
/* protection of your CA's private key.                         */
/*                                                            */
/* Supported RSA key types                                 @LHC*/
/*                                                            */
/* - Type  0: Generated using software and stored in RACF.      */
/*            Acceptable key sizes range from 512 to 4096 bits. */
/*            This is the default key type.                @LHC*/
/* - Type  2: Generated using the ICSF CCA cryptographic        */
/*            coprocessor and stored in a Public Key Data Set   */
/*            (PKDS).  Acceptable key sizes range from 512 to   */
/*            4096 bits.  This key type is not FIPS compliant   */
/*            (see the Notes below).                       @LHC*/
/* - Type  8: Generated using the ICSF PKCS#11 cryptographic    */
/*            coprocessor and stored in a Token Key Data Set    */
/*            (TKDS).  Acceptable key sizes range from 512 to   */
/*            4096 bits.                                   @LHC*/
/*                                                            */
/* Supported DSA key types                                 @LHC*/
/*                                                            */
/* - Type  3: Generated using software and stored in RACF.      */
/*            Acceptable key sizes range from 512 to 2048       */
/*            bits.                                        @LHC*/
/*                                                            */
/* Supported NIST Elliptic Curve Cryptography (ECC) key         */
/* types                                                   @LHC*/
/*                                                            */
/* - Type  4: Generated using software and stored in RACF.      */
/*            Acceptable key sizes are 192, 224, 256, 384, and  */
/*            521 bits (521 is not a typographical error).  @LHC*/
/* - Type  6: Generated using the ICSF CCA cryptographic        */
/*            coprocessor and stored in a Public Key Data Set   */
/*            (PKDS).  Acceptable key sizes are 192, 224, 256,  */
/*            384, and 521 bits.  This key type is not FIPS     */
/*            compliant (see the Notes below).             @LHC*/
/* - Type  9: Generated using the ICSF PKCS#11 cryptographic    */
/*            coprocessor and stored in a Token Key Data Set    */
/*            (TKDS).  Acceptable key sizes are 192, 224, 256,  */
/*            384, and 521 bits.                           @LHC*/
/*                                                            */
/* Supported Brainpool Elliptic Curve Cryptography (ECC) key    */
/* types                                                   @LHC*/
/*                                                            */
/* - Type  5: Generated using software and stored in RACF.      */
/*            Acceptable key sizes are 192, 224, 256, 384, and  */
/*            512 bits.  This key type is not FIPS compliant    */
/*            (see the Notes below).                       @LHC*/
/* - Type  7: Generated using the ICSF CCA cryptographic        */
/*            coprocessor and stored in a Public Key Data Set   */
/*            (PKDS).  Acceptable key sizes are 160, 192, 224,  */
/*            256, 320, 384, and 512 bits.  This key type is    */
/*            not FIPS compliant (see the Notes below).    @LHC*/
/* - Type 10: Generated using the ICSF PKCS#11 cryptographic    */
/*            coprocessor and stored in a Token Key Data Set    */
/*            (TKDS).  Acceptable key sizes are 160, 192, 224,  */
/*            256, 320, 384, and 512 bits.  This key type is    */
/*            not FIPS compliant (see the Notes below).    @LHC*/
/*                                                            */
/* Notes                                                      */
/* -----                                                      */
/* 1.  Certificates with keys stored in RACF can be backed up   */
/*     to a dataset.                                       @LHC*/
/* 2.  Key types using the ICSF CCA cryptographic coprocessor   */
/*     require ICSF to be operational and to be configured with */
/*     a PKDS.                                             @LHC*/
/* 3.  Key types using the ICSF PKCS#11 cryptographic           */
/*     coprocessor require ICSF to be operational and to be     */
/*     configured with a TKDS.                             @LHC*/
/* 4.  For a CA certificate to be FIPS compliant:               */
/*     A.  The key cannot be stored in a PKDS.                  */
/*     B.  Brainpool ECC keys cannot be used.                   */
/*     C.  Keys must meet the minimum size requirement for the  */
/*         intended FIPS level.  The FIPS level to be used by   */
/*         PKI Services is specified by the environment         */
/*         variable _PKISERV_FIPS_LEVEL:                        */
/*             ----------------+--------+--------+--------      */
/*                             | _PKISERV_FIPS_LEVEL value      */
/*                             |    1   |    2   |    3         */
/*             ----------------+--------+--------+--------      */
/*             RSA             |  1024  |  2048  |  2048        */
/*             ----------------+--------+--------+--------      */
/*             DSA             |  1024  |  2048  |  2048        */
/*             ----------------+--------+--------+--------      */
/*             NIST ECC        |   192  |   224  |   224        */
/*             ----------------+--------+--------+--------  @LHC*/
/*                                                              */
/* Select the type desired by setting key_type to one of the    */
/* above options.                                           @LHA*/
/*                                                              */
/****************************************************************/
key_type=0

/****************************************************************/
/* If you set key_type=2, 6, 7, 8, 9 or 10 above, you will      */
/* need to restrict access to the CA's private key. Unless you  */
/* indicate otherwise, this exec will activate the CSFKEYS      */
/* classes create a profile in the CSFKEYS class to protect the */
/* CA's private key, and permit the PKI Services daemon to use  */
/* it.                                                      @LHC*/
/*                                                              */
/* If you are already using ICSF, then you may have profiles in */
/* the CSFSERV class protecting ICSF services.  If you are not  */
/* already using ICSF in previous releases of PKI Servuces, you */
/* need to set up CSFOWH profile in this release and allow the  */
/* PKI Services daemon to have READ access.  If you set         */
/* key_type=2, 4, 5, 6, 7, 8, 9 or 10 above, the PKI            */
/* Services daemon would need access to the following profiles: */
/*  CSFDSV, CSFDSG, CSFPKI, CSFENC, CSFDEC                      */
/*                                                          @02C*/
/* Also, the PKI Services surrogate ID would need access        */
/* to the following profiles that cover the services used       */
/* during the SSL/TLS handshake processing depending on         */
/* the cipher and certificates being utilized:                  */
/*  CSFENC, CSFDEC, CSFPKD, CSFPKE, CSFPKI, CSFDSG, CSFDSV,     */
/*  CSFCKI, CSFCKM, CSF1DVK, CSF1GAV, CSF1GKP, CSFIQA, CSF1PKS, */
/*  CSF1PKV, CSF1TRD, CSF1TRC                                   */
/*                                                          @D8C*/
/* You may already have a RACF group for authorized ICSF users. */
/* If you specify the group name for csfusers_grp, the daemon   */
/* and surrogate IDs will be connected to the group, and the    */
/* group permitted to the profiles. Otherwise the two IDs will  */
/* be directly permitted to the profiles.                       */
/*                                                          @D8C*/
/* Set the following variables as needed:                       */
/*                                                              */
/* csfkeys_profile - Profile to be created in the CSFKEYS class */
/*     Set the value to '' if you don't want the profile        */
/* csfserv_profile - Profile to be created in the CSFSERV class */
/*     e.g., 'CSF*'                                             */
/* csfusers_grp - Group name for authorized ICSF users          */
/*     e.g., 'ICSFUGRP'                                         */
/****************************************************************/
csfkeys_profile='IRR.DIGTCERT.CERTIFAUTH.*'

if (key_type=2 | key_type=4 |,
    key_type=5 | key_type=6 | key_type=7 |,
    key_type=8 | key_type=9 | key_type=10 |,           /* @LHC */
    key_gen=1)
  then csfserv_profile='CSF*'
  else csfserv_profile='CSFOWH'                        /* @02C */

csfusers_grp=''

/****************************************************************/
/* Question 4 - Back up your private key?                       */
/*                                                              */
/* The exec will prompt you to enter a pass phrase to encrypt a */
/* backup copy of your CA's certificate and private key.        */
/* Caution, the text you enter at the prompt WILL be displayed  */
/* at the terminal. Backup is highly recommended.  If you do not*/
/* wish to back up your CA's certificate and private key to a   */
/* pass phrase encrypted data set, set key_backup=0. The back up*/
/* may be done later if the key is not stored in ICSF.          */
/*                                                            */
/* Note, back up is not performed if the CA certificate was not */
/* created by this exec or if you specified key_type 2, 6, 7, 8,*/
/* 9 or 10 above.                                         @LEC*/
/* ************************************************************/
key_backup=1

/**************************************************************/
/* Question 5 - Set up z/OS UNIX level security?              */
/*                                                            */
/* z/OS UNIX may be set up to operate with a higher level of  */
/* security than traditional UNIX. While we recommend this, it */
/* difficult to set up. You may want to defer this until later. */
/*                                                            */
/* If you don't want to set up UNIX security now, leave       */
/* unix_sec=0.                                                */
/*                                                            */
/* If you already have UNIX level security established and wish */
/* to continue it, set unix_sec=1.                            */
/*                                                            */
/* If you don't have UNIX level security established and wish  */
/* to establish it now, set unix_sec=2. Note additional manual  */
/* configuration probably will be required. This can be done   */
/* by adding, removing, updating members of the two stem       */
/* variables below. The pgmcntl_dsn stem contains the data set  */
/* names of load libraries that need program control. The      */
/* bpx_userid stem contains the user IDs of your server daemons.*/
/* (These need access to BPX.SERVER and BPX.DAEMON in the      */
/* FACILITY class.) Again, you can defer this until later by   */
/* leaving unix_sec=0                                          */
/**************************************************************/
unix_sec=0
pgmcntl_dsn.0=8  /* Number of program controlled data sets below @L2C*/
pgmcntl_dsn.1="'CEE.SCEERUN'"
pgmcntl_dsn.2="'CBC.SCLBDLL'"
pgmcntl_dsn.3="'SYS1.SIEALNKE'"  /* Common LINKLIST PDSE dataset @L2A*/
pgmcntl_dsn.4="'SYS1.CSSLIB'"                                /* @L2C*/
pgmcntl_dsn.5="'TCPIP.SEZALOAD'"                             /* @L2C*/
pgmcntl_dsn.6="'SYS1.LINKLIB'"                               /* @L2C*/
pgmcntl_dsn.7="'CSF.SCSFMOD0'"                               /* @L2C*/
pgmcntl_dsn.8="'CSF.SCSFMOD1'"                               /* @L2C*/
bpx_userid.0=1 /* Number of additional bpx server ids below */
bpx_userid.1="OMVSKERN"

/**************************************************************/
/* Question 6 -  Use DB2 as the repository for the Issued      */
/*               Certificate List (ICL) and Object Store?      */
/*                                                            */
/* In the default configuration, PKI Services uses VSAM data   */
/* sets as the repository for the issued certificate list (ICL) */
/* and the object store.  VSAM is an included feature of z/OS.  */
/*                                                            */
/* Optionally, DB2 for z/OS may be used as the repository for   */
/* the ICL and the object store.  DB2 for z/OS is a separately  */
/* purchased and installed product which provides additional    */
/* data stability features and query capabilities beyond those  */
/* provided by VSAM.  PKI Services may use DB2 for z/OS as the  */
/* repository for the ICL and object store if DB2 is running on */
/* the same system as the PKI Services daemon.                 */
/*                                                            */
/* If you wish to use DB2 for z/OS as the repository for the    */
/* ICL and object store, set db2_repos to 1; otherwise, leave   */
/* db2_repos set to 0.                                        */
/*                                                       @LCA */
/**************************************************************/
db2_repos = 0                                           /* @LCA */

/**************************************************************/
/* If you set db2_repos = 1 above, this exec will permit the    */
/* PKI Services daemon access to the DB2 Resource Recovery      */
/* Services access facility (RRSAF).                           */
/*                                                            */
/* To grant this access, the name of the local DB2 subsystem    */
/* that will be used to provide the repository must be         */
/* provided.  This is a name of up to 4 characters.  You can    */
/* obtain this name from your DB2 for z/OS systems programmer.  */
/* Change the value of db2_subsys below to contain the name of  */
/* the local DB2 subsystem.                                   */
/*                                                       @LCA */
/**************************************************************/
db2_subsys = 'DSN9'                                     /* @LCA */
```

```
/*************************************************************/
/* Question 7 - Need granular control on administrative      */
/*              functions?                                   */
/* If you wish to set up granular control, set               */
/* AdminGranularControl to 1; otherwise, leave               */
/* AdminGranularControl set to 0.                            */
/*************************************************************/
AdminGranularControl = 0                               /* @DAC */

/*************************************************************/
/* If you set AdminGranularControl to 1,                     */
/* provide the template nick names you want to act on and    */
/* assign the groups.                                        */
/* Change the value of template1,2,3 below and add more if   */
/* needed.                                                   */
/* Change the value of pkigroup1,2 below and add more if     */
/* needed.                                                   */
/*************************************************************/
template.0 = 12                                        /* @LDA */
template.1 = "1YBSSL"                                  /* @LDA */
template.2 = "1YBSM"                                   /* @LDA */
template.3 = "2YBWL"                                   /* @LDA */
template.4 = "2YBZOS"                                  /* @LDA */
template.5 = "5YSSSL"                                  /* @LDA */
template.6 = "5YSIPS"                                  /* @LDA */
template.7 = "5YSCAP"                                  /* @LDA */
template.8 = "2YIACS"                                  /* @LDA */
template.9 = "SAMPLB"                                  /* @LDA */
template.10= "5YSCEPP"                                 /* @LDA */
template.11= "1YKRC"                                   /* @LDA */
template.12= "2YEVSSL"                                 /* @LDA */

pkigroup1="PKIGRP1"       /* PKI Services Admin group name @LDA */
pki_gid1="656"              /* PKI Services Admin group id @LDA */
pkigroup1_mem.0=0 /* Number of pkigroup members to connect @LDA */
pkigroup1_mem.1=""                                     /* @LDA */

actions1.0=13                                          /* @DAC */
actions1.1="QUERYREQS"                                 /* @LDA */
actions1.2="QUERYCERTS"                                /* @LDA */
actions1.3="APPROVE"                                   /* @LDA */
actions1.4="APPROVEWITHMODS"                           /* @LDA */
actions1.5="REJECT"                                    /* @LDA */
actions1.6="DELETEREQS"                                /* @LDA */
actions1.7="REVOKE"                                    /* @LDA */
actions1.8="DELETECERTS"                               /* @LDA */
actions1.9="RESUME"                                    /* @LDA */
actions1.10="AUTORENEWENABLE"                          /* @LDA */
actions1.11="AUTORENEWDISABLE"                         /* @LDA */
actions1.12="CHANGEMAIL"                               /* @LDA */
actions1.13="PREREGISTER"                              /* @LDA */
                                                       /* 1@DAD*/
pkigroup2="PKIGRP2"       /* PKI Services Admin group name @LDA */
pki_gid2="657"              /* PKI Services Admin group id @LDA */
pkigroup2_mem.0=0 /* Number of pkigroup members to connect @LDA */
pkigroup2_mem.1=""                                     /* @LDA */

actions2.0=2                                           /* @LDA */
actions2.1="QUERYREQDETAILS"                           /* @LDA */
actions2.2="QUERYCERTDETAILS"                          /* @LDA */

/*-----------------------------------*/
/* Part 3 - Things you can change  */
/*-----------------------------------*/

/*************************************************************/
/* Label of the CA certificate that is the superior (signer) of */
/* the PKI Services CA, if self sign leave blank             */
/*************************************************************/
signing_ca_label = ""                                    /*@L4A*/

/*************************************************************/
/* This exec will record results to a log data set if desired.  */
/* the name of the data set is specified below. If you do not   */
/* want log data set recording, set log_dsn="" (Not recommended)*/
/*************************************************************/
if (ca_domain = "") then          /* If no CA Domain...     @L4A*/
   log_dsn="IKYSETUP.LOG"         /* Under your ID             */
else                              /* Else use CA Domain     @L4A*/
   log_dsn=ca_domain_trunc||".IKYSETUP.LOG" /* CA Domain qualified @L4A*/

/*************************************************************/
```

```
/* Note IKYCVSV1, the sample JCL to create VSAM datasets and    */
/* pkiserv.conf expect the object store and ICL datasets to     */
/* have PKISRVD as their high level qualifier.                  */
/* Changing either "daemon" or "vsamhlq" will                   */
/* require making the same change to IKYCVSV1 and               */
/* pkiserv.conf                                             @LIC*/
/****************************************************************/
vsamhlq=daemon     /* HLQ for VSAM data sets. Same as daemon ID */

/****************************************************************/
/* The following variables are used to create the certificates  */
/* for the Certificate Authority (CA) and the web server. If    */
/* the default settings are used, these certificates will       */
/* expire as follows:                                           */
/*    - The CA certificate will expire 20 years from the date   */
/*      when this exec is run.                                  */
/*    - The web server SSL certificate will expire 5 years from */
/*      the date when this exec is run.                         */
/*                                                              */
/* You may tailor the expiration dates to suit your own need in */
/* one of two ways, which are described below.  If you decide   */
/* to tailor the expiration dates for these certificates, make  */
/* sure that the web server SSL certificate's expiration date   */
/* does not exceed the expiration date of the CA certificate;   */
/* failure to ensure this will result in the web server SSL     */
/* certificate being added as NOTRUST in the RACF database.     */
/*    - Method 1: You can change the number of years used to    */
/*      set the CA certificate and web server SSL certificate.  */
/*      If you choose this method, leave the code segment       */
/*      labelled "Method 2" below commented out of the exec,    */
/*      and alter the values for the "ca_exyears" and           */
/*      "web_exyears" to set the number of years for the CA     */
/*      certificate and the web server SSL certificate,         */
/*      respectively.  Ensure that the value for "ca_exyears"   */
/*      equals or exceeds the  value for "web_exyears".         */
/*    - Method 2: You can hardcode the expiration date to be    */
/*      used for the CA certificate and the web server SSL      */
/*      certificate.   If you choose this method, comment out   */
/*      the code segment labelled "Method 1" below, uncomment   */
/*      the code segment labelled "Method 2" below, and alter   */
/*      the values for the "ca_expires" and "web_expires" to    */
/*      set the expiration dates for the CA certificate and the */
/*      web server SSL certificate, respectively.  Ensure that  */
/*      the expiration date given in "ca_expires" is later      */
/*      than the date specified in "web_expires".               */
/******************************************************** @D9A */

/*
   Method 1: Set the CA certificate and web server
   certificate expiration dates to a number of years in
   the future.  If using Method 2 to set the expiration
   dates, comment out the code in this section.             @D9A
 */
ca_exyears  = 20                                        /* @D9C */
web_exyears = 5                                         /* @D9C */

if (datatype(ca_exyears,'N')^=1) then do
  say "The CA certificate expiration years value",
      ca_exyears "is not valid."                        /* @D9A */
  return 8                                              /* @D9A */
end                                                     /* @D9A */
if (datatype(web_exyears,'N')^=1) then do
  say "The webserver SSL certificate expiration years value",
      web_exyears "is not valid."                       /* @D9A */
  return 8                                              /* @D9A */
end

today = date('S')                                       /* @D9A */

exday = ca_exyears * 10000 + today                      /* @D9A */
ca_expires = substr(exday,1,4)"/"||,
             substr(exday,5,2)"/"||,
             substr(exday,7,2)                          /* @D9A */

exday = web_exyears * 10000 + today                     /* @D9A */
web_expires = substr(exday,1,4)"/"||,
              substr(exday,5,2)"/"||,
              substr(exday,7,2)                         /* @D9A */
/* End of Method 1.                                        @D9A */

/*
   Method 2: Set the CA certificate and web server
   certificate expiration dates to a specific future
   date.  If using Method 1 to set the expiration
   dates, leave this code segment commented out of
   the exec.  If using this method, remove the
   comment characters from the following two
   instructions, change the dates to the desires
   values if necessary, and comment out all
   instructions in Method 1.                                   @D9A
 */
/* ca_expires  = "2030/01/01" */                        /* @D9A */
/* web_expires = "2015/01/01" */                        /* @D9A */
/* End of Method 2.                                        @D9A */

web_label = "SSL Cert"            /* Label for web server cert  */

/***************************************************************/
/* PKI Services Key ring name                                  */
/***************************************************************/
if (ca_domain = "") then          /* If no CA Domain...     @L4A*/
   ca_ring="CAring"               /* keyring name for PKI Srvs  */
else                              /* Else use CA Domain     @L4A*/
   ca_ring="CAring."||ca_domain   /* CA Domain qualified    @L4A*/

/***************************************************************/
/* PKI Services PKCS11 Token Name                              */
/***************************************************************/
if (ca_domain = "") then          /* If no CA Domain...     @LEA*/
   caStore=daemon||".CATOKEN"      /* token name for PKI Srvs   */
else                              /* Else use CA Domain     @DBC*/
   caStore=daemon||".CATOKEN." || ca_domain
                                  /* CA Domain qualified    @DBC*/

/***************************************************************/
/* You can select the size (in bits) of your CA's private key.  */
/* Acceptable values depend upon the value that is used for     */
/* "key_type" and other system conditions.  Refer to the        */
/* instructions given Part 2 "Questions you must answer",       */
/* Question 3  "What key type to use for the CA certificate?"   */
/* for advice on selecting the appropriate size.            @LHC*/
/*                                                              */
/* The default value to use for the key size is given in the    */
/* "ca_keysize" value below.  This default value has been       */
/* chosen with the assumption that an RSA key will be used in   */
/* the CA certificate, and would result in an RSA key that is   */
/* compliant with any FIPS level used by PKI Services.      @LHC*/
/***************************************************************/
ca_keysize="2048"

/***************************************************************/
/* Data set to contain the backup copy of the CA certificate    */
/* and private key. (pass phrase encrypted PKCS#12 format)      */
/***************************************************************/
if (ca_domain = "") then    /* If no CA Domain...           @L4A*/
   backup_dsn = "'"||daemon||".KEY.BACKUP.P12BIN'"       /* @L5C*/
else                        /* Else use CA Domain           @L4A*/
   backup_dsn = "'"||daemon||"."||ca_domain_trunc||,
                ".KEY.BACKUP.P12BIN'"                    /* @L5C*/
                            /* CA Domain qualify backup dsn @L4A*/

/***************************************************************/
/* Data set to contain the exported copy of the CA certificate  */
/* (DER encoded). This is to assist the backup process.     @L9C*/
/*                                                              */
/***************************************************************/
if (ca_domain = "") then    /* If no CA Domain...           @L4A*/
   cacert_dsn = "'"||daemon||".CACERT.DERBIN'"
else                        /* Else use CA Domain           @L9C*/
   cacert_dsn = "'"||daemon||"."||ca_domain_trunc||".CACERT.DERBIN'"
                            /* CA Domain qualify export dsn @L9C*/

/***************************************************************/
/* Data set to contain the exported copy of the webserver's     */
/* root certificate (DER encoded). This is to be OPUT to an     */
/* HFS file later to enable easy downloading by clients.    @L9A*/
/***************************************************************/
if (ca_domain = "") then    /* If no CA Domain...           @L9A*/
   export_dsn = "'"||daemon||".WEBROOT.DERBIN'"
else                        /* Else use CA Domain           @L9A*/
   export_dsn = "'"||daemon||"."||ca_domain_trunc||".WEBROOT.DERBIN'"
                            /* CA Domain qualify export dsn @L9A*/

/***************************************************************/
```

```
/* Data set to contain the backup copy of the RA certificate    */
/* and private key. (pass phrase encrypted PKCS#12 format)      */
/****************************************************************/
if (ca_domain = "") then    /* If no CA Domain...           @01A*/
   ra_backup_dsn = "'"||daemon||".RAKEY.BACKUP.P12BIN'"  /* @L5C*/
else                        /* Else use CA Domain           @01A*/
   ra_backup_dsn = "'"||daemon||"."||ca_domain_trunc||,
                   ".RAKEY.BACKUP.P12BIN'"              /* @L5C*/
                         /* CA Domain qualify RA backup dsn @01A*/

/****************************************************************/
/* This EXEC expects the web server to be set up.  If this is   */
/* not the case, please refer to:                               */
/* z/OS HTTP Server Planning, Installing and Using.             */
/* If the user ID assigned to the IBM HTTP Server Daemon is not */
/* WEBSRV, please update the assignment below.                  */
/****************************************************************/
webserver="WEBSRV"

/*--------------------------------------------------------------*/
/* End of configurable section                                  */
/*--------------------------------------------------------------*/

parse upper arg "RUN(" runopt ")"

if runopt = '' then
  runopt="NO"
if runopt ^= "YES" & runopt ^= "PROMPT" & runopt ^= "NO" then do
  say "syntax ex 'data-set-name(IKYSETUP)' 'run(yes | no | prompt)'"
  return 8
end
if runopt ^= "YES" & runopt ^= "PROMPT" then
  runopt="NO"

say 'IKYSETUP EXEC invoked ...'
return_code= '0'
max_return_code= '0'
logdata.0=0

if log_dsn ^= "" then do
  say "Allocating log data set" log_dsn "..."
  x = OUTTRAP(MSGS.)
  "FREE FI(IKYLOGDD)"
  "FREE DA("||log_dsn||")"
  "DELETE" log_dsn
  x = OUTTRAP('OFF')
  "ALLOCATE DA("||log_dsn||") FILE(IKYLOGDD) RECFM(V B)" ,
  " LRECL(256) DSORG(PS) BLKSIZE(2560) SP(1,1) TRACKS "
  al_rc= rc
  IF al_rc ^= 0 THEN
    do
      say 'Allocation of log data set failed.'
      return 8
    end
end

call logsay "RUN("runopt") requested on" DATE() 'at' TIME() '...'
if runopt="NO" then
  call logsay "Running in test mode. Commands are not being invoked"


/******************************************************************/
/* Verify the requested key size, based upon the selected key     */
/* key type.                                                      */
/*    - When key_type=0 (RSA software generated key) or           */
/*      key_type=2 (RSA PCICC generated key), the acceptable      */
/*      range is 512 to 4096.                                     */
/*    - When key_type=3 (DSA software generated key), the         */
/*      acceptable value range is 512 to 2048.                @02C*/
/*    - When key_type=4 (NIST ECC software generated key),    @LBC*/
/*        or key_type=6 (NIST ECC key in PKDS), the           @LBA*/
/*        or key_type=9 (NIST ECC key in TKDS), the           @LEA*/
/*      acceptable set of values are 192, 224, 256, 384, and 521. */
/*    - When key_type=5 (Brainpool ECC software generated key),   */
/*        or key_type=7 (Brainpool ECC in PKDS),              @LBA*/
/*        or key_type=10 (Brainpool ECC in TKDS),             @LEA*/
/*      the acceptable set of values are 160, 192, 224, 256, 320, */
/*      384, and 512.                                             */
/*    - When key_type=8 (RSA in TKDS), the acceptable range is    */
/*      1024 to 4096.                                         @LEA*/
/* If invalid key type or size is chosen, we will issue message   */
/* and end execution.  Because this module is unable to determine */
/* the FIPS level of PKI Services, the key type and size are not  */
/* tested for their FIPS compliance.                        @LHC*/
/****************************************************************/
if (key_type ^= 0 & key_type ^= 2 & key_type ^= 3 &,
    key_type ^= 4 & key_type ^= 5 & key_type ^= 6 &,
    key_type ^= 7 & key_type ^= 8 & key_type ^= 9 &,
    key_type ^= 10)                                          /*@LHA*/
then do                                                      /*@LHA*/
    call logsay "Unsupported key type" key_type              /*@LHA*/
    return 8                                                 /*@LHA*/
end                                                          /*@LHA*/
select                                                       /*@LAA*/
  when key_type = 0 then do                                  /*@LAA*/
    if ca_keysize < 512                                      /*@LAA*/
    then do                                                  /*@LAA*/
      call logsay "Key size cannot be less than 512",
                  "for key type" key_type                    /*@LAA*/
      return 8                                               /*@LAA*/
    end                                                      /*@LAA*/
    else do                                                  /*@LAA*/
      if ca_keysize > 4096                                   /*@LAA*/
      then do                                                /*@LAA*/
        call logsay "Key size cannot be greater than",
                    "4096 for key type" key_type             /*@LAA*/
        return 8                                             /*@LAA*/
      end                                                    /*@LAA*/
    end                                                      /*@LAA*/
  end                                                        /*@LAA*/
                                                           /*14@LHD*/
  when key_type = 2 then do                                  /*@LAA*/
    if ca_keysize < 512                                      /*@LAA*/
    then do                                                  /*@LAA*/
      call logsay "Key size cannot be less than 512",
                  "for key type" key_type                    /*@LAA*/
      return 8                                               /*@LAA*/
    end                                                      /*@LAA*/
    else do                                                  /*@LAA*/
      if ca_keysize > 4096                                   /*@LAA*/
      then do                                                /*@LAA*/
        call logsay "Key size cannot be greater than",
                    "4096 for key type" key_type             /*@LAA*/
        return 8                                             /*@LAA*/
      end                                                    /*@LAA*/
    end                                                      /*@LAA*/
  end                                                        /*@LAA*/
  when key_type = 3 then do                                  /*@LAA*/
    if ca_keysize < 512                                      /*@LAA*/
    then do                                                  /*@LAA*/
      call logsay "Key size cannot be less than 512",
                  "for key type" key_type                    /*@LAA*/
      return 8                                               /*@LAA*/
    end                                                      /*@LAA*/
    else do                                                  /*@LAA*/
      if ca_keysize > 2048                                   /*@LFC*/
      then do                                                /*@LAA*/
        call logsay "Key size cannot be greater than",
                    "2048 for key type" key_type             /*@LFC*/
        return 8                                             /*@LAA*/
      end                                                    /*@LAA*/
    end                                                      /*@LAA*/
  end                                                        /*@LAA*/
  when key_type = 4 |,                                       /*@LBC*/
       key_type = 6 |,                                       /*@LEC*/
       key_type = 9 then do                                  /*@LEC*/
    if ca_keysize ^= 192 & ca_keysize ^= 224 &,
       ca_keysize ^= 256 & ca_keysize ^= 384 &,
       ca_keysize ^= 521                                     /*@LAA*/
    then do                                                  /*@LAA*/
      call logsay "Key size must be: 192, 224, 256, 384,",
                  "or 521 for key type" key_type             /*@LAA*/
      return 8                                               /*@LAA*/
    end                                                      /*@LAA*/
  end                                                        /*@LAA*/
  when key_type = 5 |,                                       /*@LBC*/
       key_type = 7 |,                                       /*@LEC*/
       key_type = 10 then do                                 /*@LEC*/
    if ca_keysize ^= 160 & ca_keysize ^= 192 &,
       ca_keysize ^= 224 & ca_keysize ^= 256 &,
       ca_keysize ^= 320 & ca_keysize ^= 384 &,
       ca_keysize ^= 512                                     /*@LAA*/
    then do                                                  /*@LAA*/
      call logsay "Key size must be: 160, 192, 224, 256,",
                  "320, 384, or 512 for key type",
                  key_type                                  /*@LAA*/
      return 8                                              /*@LAA*/
    end                                                     /*@LAA*/
  end                                                       /*@LAA*/
  when key_type = 8 then do                                 /*@LEA*/
    if ca_keysize < 1024                                    /*@LEA*/
    then do                                                 /*@LEA*/
      call logsay "Key size cannot be less than 1024",
                  "for key type" key_type                   /*@LEA*/
      return 8                                              /*@LEA*/
    end                                                     /*@LEA*/
    else do                                                 /*@LEA*/
      if ca_keysize > 4096                                  /*@LEA*/
      then do                                               /*@LEA*/
        call logsay "Key size cannot be greater than",
                    "4096 for key type" key_type            /*@LEA*/
        return 8                                            /*@LEA*/
      end                                                   /*@LEA*/
    end                                                     /*@LEA*/
  end                                                       /*@LEA*/
  otherwise do                                              /*@LAA*/
    call logsay "Key type must be: 0 through 10."           /*@LEC*/
    return 8                                                /*@LAA*/
  end                                                       /*@LAA*/
end                                                         /*@LAA*/
                                                          /*10@LAD*/

/*******************************************************************/
/* Create the daemon and surrogate user IDs using RACF ADDUSER TSO*/
/* command. Give them an OMVS segment since they will need access */
/* to UNIX System Services.                                        */
/*******************************************************************/


call logsay2 "Creating users and groups ..."
call tsoserv "ADDUSER " daemon "name('PKI Srvs Daemon')",
  " nopassword",
  " omvs(uid("daemon_uid")",
  " assize(256000000)",
  " threads(512))"

if restrict_surrog=1 then   /*@D1C*/
  resattr="restricted"
else
  resattr=""
call tsoserv "ADDUSER " surrog "nopassword",
  resattr,
  " omvs(uid("surrog_uid"))",
  " name('PKI Srvs Surrogate')"

/*****************************************************************/
/* Set up PKI Services administration group.                     */
/* If AdminGranularControl = 1, add additional administration    */
/* groups for granular control as specified for Question 7.      */
/*****************************************************************/
if pkigroup ^= "" then do                                     /* @LCM*/
  call tsoserv "ADDGROUP " pkigroup "OMVS(GID("pki_gid"))"    /* @LCM*/
  do i = 1 to pkigroup_mem.0                                  /* @LCM*/
    call tsoserv "CONNECT" pkigroup_mem.i "GROUP("pkigroup")" /* @LCM*/
  end                                                         /* @LCM*/
end                                                           /* @LCM*/

if AdminGranularControl=1 then do                       /* @LDA */
   if pkigroup1 ^= "" then do                           /* @LDA */
      call tsoserv "ADDGROUP " pkigroup1,               /* @LDA */
                   "OMVS(GID("pki_gid1"))"              /* @LDA */
      do i = 1 to pkigroup1_mem.0                       /* @LDA */
         call tsoserv "CONNECT" pkigroup1_mem.i,        /* @LDA */
                      "GROUP("pkigroup1")"              /* @LDA */
      end                                               /* @LDA */
   end                                                  /* @LDA */

   if pkigroup2 ^= "" then do                           /* @LDA */
      call tsoserv "ADDGROUP " pkigroup2,               /* @LDA */
                   "OMVS(GID("pki_gid2"))"              /* @LDA */
      do i = 1 to pkigroup2_mem.0                       /* @LDA */
         call tsoserv "CONNECT" pkigroup2_mem.i,        /* @LDA */
                      "GROUP("pkigroup2")"              /* @LDA */
      end                                               /* @LDA */
   end                                                  /* @LDA */
end                                                     /* @LDA */
```

```
/*****************************************************************/
/* Set up the permission needed for the DB2 tables or the VSAM  */
/* data sets.                                               @LCA*/
/*****************************************************************/

/****************************************************************
 * The DB2 for z/OS Resource Recovery Services access facility
 * (RRSAF) is protected by the RRSAF DSNR class resource that
 * is specific to the DB2 subsystem.  The following command
 * gives READ access to the PKI Services daemon, which allows
 * the daemon the capability of interacting with the DB2
 * subsystem through this access facility.                 @LCA
 ****************************************************************/
if db2_repos=1 then do                                        /* @LCA*/
   call logsay2 "Granting DB2 RRSAF access to the PKI",
                "Services daemon."                            /* @LCA*/
   call tsoserv "RDEFINE DSNR "db2_subsys".RRSAF",
                "UACC(NONE)"                                  /* @LCA*/
   call tsoserv "PERMIT "db2_subsys".RRSAF",
                "CLASS(DSNR) ACCESS(READ)",
                "ID("daemon")"                                /* @LCA*/
   call tsoserv "SETROPTS CLASSACT(DSNR)"                     /* @LCA*/
end                                                           /* @LCA*/
/***************************************************************
 * The VSAM data sets created for PKI Services are protected.
 * Grant access to these data sets for the PKI Services daemon
 * and the administrator.                                  @LCA
 ***************************************************************/
else do                                                       /* @LCA*/
   call tsoserv "SETROPTS EGN GENERIC(DATASET)"

   call tsoserv "ADDSD '"vsamhlq".**' UACC(NONE)"
   call tsoserv "PERMIT '"vsamhlq".**' ID("daemon")",
                "ACCESS(ALTER)"                               /* @D4A*/

   if (vsamhlq ^= daemon) then do                             /* @D4A*/
     call tsoserv "ADDSD '"daemon".**' UACC(NONE)"            /* @D4A*/
     call tsoserv "PERMIT '"daemon".**' ID("daemon")",
                  "ACCESS(ALTER)"                             /* @D4A*/
   end                                                        /* @D4A*/

   /************************************************************
    * Give the administrators access to the VSAM data sets
    * identified in the [ObjectStore] section of
    * the pkiserv.conf file.
    ************************************************************/
    call logsay2 "Allowing administrators to access",
                 "PKI VSAM databases ..."                     /* @LCC*/
    call tsoserv "PERMIT '"vsamhlq".**' ID("pkigroup")",
                 "ACCESS(CONTROL)"
    call tsoserv "SETROPTS GENERIC(DATASET) REFRESH"
end                                                           /* @LCA*/

                                                              /*6@LCM*/
/*********************************************************************/
/* If the key_type is 8 or 9 or 10 a PKCS #11 token must be created  */
/* first before using it to create a CA certificate with Secure Key  */
/* in the TKDS.                                                  @LEA*/
/*********************************************************************/
if (key_type=8 | key_type=9 | key_type=10) then do      /* @LEA */
   call logsay2 "Creating a PKCS #11 token... "         /* @LEA */
   call tsoserv "RACDCERT ADDTOKEN("caStore")"          /* @LEA */
end                                                     /* @LEA */

/*****************************************************************/
/* In order to create and sign digital certificates for others  */
/* you need to define or import in RACF a Certificate Authority */
/* certificate and associated private key.                      */
/* This is done using the RACF RACDCERT GENCERT command.        */
/*****************************************************************/
if ca_dn ^= "" then do
  call logsay2 "Creating the CA certificate ..."
  if signing_ca_label = "" then                         /*@L4A*/
    certcmd = "RACDCERT GENCERT CERTAUTH SUBJECTSDN("ca_dn")",
     " WITHLABEL('"ca_label"') NOTAFTER(DATE("ca_expires"))",
     " SIZE("ca_keysize")"
   else                                                 /*@L4A*/
    certcmd = "RACDCERT GENCERT CERTAUTH SUBJECTSDN("ca_dn")",
     " WITHLABEL('"ca_label"')",
     " SIGNWITH(CERTAUTH LABEL('"signing_ca_label"'))",
     " NOTAFTER(DATE("ca_expires"))",
     " SIZE("ca_keysize")"                              /*@L4A*/
```

```
                                                              /*@LHD*/
  if key_type=2 then                                          /*@LHC*/
    certcmd= certcmd || " PCICC"
  else if key_type=3 then
    certcmd =certcmd || " DSA"
  else if key_type=4 then
    certcmd =certcmd || " NISTECC"                            /*@D5A*/
  else if key_type=5 then
    certcmd =certcmd || " BPECC"                              /*@D6A*/
  else if key_type=6 then
    certcmd =certcmd || " NISTECC(PKDS)"                      /*@LBA*/
  else if key_type=7 then
    certcmd =certcmd || " BPECC(PKDS)"                        /*@LBA*/
  else if key_type=8 then
    certcmd =certcmd || " RSA(TOKEN("caStore"))"              /*@LEA*/
  else if key_type=9 then
    certcmd =certcmd || " NISTECC(TOKEN("caStore"))"          /*@LEA*/
  else if key_type=10 then
    certcmd =certcmd || " BPECC(TOKEN("caStore"))"            /*@LEA*/

    /* Add Subject Altname extension for EST CA cert */
  ca_altname = ""

  if (ca_altdomain ^= "") then
    ca_altname = ca_altname||,
    ca_altdomain

  if (ca_altip ^= "") then
    ca_altname = ca_altname||,
    ca_ip


  if ca_altname ^= "" then
    certcmd =certcmd || "ALTNAME("ca_altname")"


  call tsoserv certcmd

  if key_type^=2 & key_type^=6 & key_type^=7 & key_type^=8,
     & key_type^=9 & key_type^=10 & key_backup=1        /* @LEC*/
    then do                                             /* @D8C*/
/*****************************************************************/
/* Export certificate and key to PKCS#12 dataset                 */
/*****************************************************************/
    say ""
    say "Enter a passphrase to protect the key. You will need"
    say " this value later if you need to restore the key."
    say ""
    say "Attention, the value will be displayed in the screen:"
    parse pull pp
    call logsay2 "Backing up the CA certificate ..."
    certcmd = "RACDCERT CERTAUTH EXPORT(LABEL('"ca_label"'))",
     " DSN("backup_dsn") FORMAT(PKCS12DER)",
     " PASSWORD('"pp"')"

    call tsoserv certcmd
  end

                                                        /*5@LHD*/
end /* ca_dn ^= "" */
/*****************************************************************/
/* Mark the CA certificate as HIGHTRUST so HostIdMappings        */
/* are honored                                                   */
/*****************************************************************/
call logsay2 "Marking CA certificate as HIGHTRUST ..."
certcmd = "RACDCERT CERTAUTH ALTER(LABEL('"ca_label"')) HIGHTRUST"
call tsoserv certcmd

/******************************************************************/
/* The CA certificate must be saved to a data set to assist the  */
/* backup process.                                          @L9C  */
/******************************************************************/
call logsay2 "Saving the CA certificate to a data set ..."/* @L9C*/
certcmd = "RACDCERT CERTAUTH EXPORT(LABEL('"ca_label"'))",
  " DSN("cacert_dsn") FORMAT(CERTDER)"                   /* @L9C*/
call tsoserv certcmd

if (ra_label ^= "") then do                              /* @L4A*/
   /***************************************************************/
   /* Creating RA Certificate                                     */
   /***************************************************************/
   call logsay2 "Creating the RA certificate ..."           /* @L4A*/
```

```
    certcmd = "RACDCERT ID("daemon") GENCERT SUBJECTSDN("ra_dn")",
      " KEYUSAGE(HANDSHAKE) SIGNWITH(CERTAUTH LABEL('"ca_label"'))",
      " NOTAFTER(DATE("ca_expires")) WITHLABEL('"ra_label"')" /* @L4A*/
    call tsoserv certcmd                                      /* @L4A*/

    /**************************************************************/
    /* Backing up RA Certificate                                  */
    /**************************************************************/
    call logsay2 "Backing up RA certificate ..."              /* @L4A*/
    certcmd = "RACDCERT ID("daemon") EXPORT(LABEL('"ra_label"'))",
      " DSN("ra_backup_dsn") FORMAT(PKCS12DER)",
      " PASSWORD('"pp"')"                                     /* @L4A*/
    call tsoserv certcmd                                      /* @L4A*/
 end                                                          /* @L4A*/


 /*****************************************************************/
 /* The CA/RA certificate must be placed in a key ring so that    */
 /* PKI Services can access it.                                   */
 /*****************************************************************/
 call logsay2 "Creating the PKI Services keyring ..."
 call tsoserv "RACDCERT ADDRING("ca_ring") ID("daemon")"
 call tsoserv "RACDCERT ID("daemon") CONNECT(CERTAUTH",
  " LABEL('"ca_label"')",
  " RING("ca_ring") USAGE(PERSONAL) DEFAULT) "
 if (ra_label ^= "") then                                      /* @L4A*/
    call tsoserv "RACDCERT ID("daemon") CONNECT(LABEL('"ra_label"')",
    " RING("ca_ring") USAGE(PERSONAL))"                        /* @D2C*/

 /*******************************************************************/
 /* Create the certificate for the webserver signed by your new CA  */
 /*******************************************************************/

 if web_dn ^= "" then do
   call logsay2 "Creating the Webserver SSL certificate and keyring ..."
   certcmd = "RACDCERT GENCERT ID("webserver") SIGNWITH(CERTAUTH",
    " LABEL('"ca_label"'))",
    " WITHLABEL('"web_label"') SUBJECTSDN("web_dn")",
    " NOTAFTER(DATE("web_expires"))"

 /* Add Subject Altname extension for web server cert */
   web_altname = ""
   if (web_altdomain ^= "") then
     web_altname = web_altname||,
     web_altdomain

   if (web_altip ^= "") then
     web_altip = web_altname||,
     web_altip

   if web_altname ^= "" then
     certcmd =certcmd || "ALTNAME("web_altname")"

   call tsoserv certcmd

 /*****************************************************************/
 /* Add the webserver's certificate to the webserver's RACF (SAF)*/
 /* key ring                                               @L9C */
 /*****************************************************************/
   call tsoserv "RACDCERT ADDRING("web_ring") ID("webserver")"
   call tsoserv "RACDCERT  ID("webserver") CONNECT(ID("webserver")",
     " LABEL('"web_label"') RING("web_ring") USAGE(PERSONAL) DEFAULT)"
 end /* web_dn ^= "" */

 /*****************************************************************/
 /* Add the PKI CA certificate to the webserver's RACF (SAF)     */
 /* key ring                                               @L9C */
 /*****************************************************************/
 if web_ring ^= "" then
   call tsoserv "RACDCERT  ID("webserver") CONNECT(CERTAUTH",
      " LABEL('"ca_label"') RING("web_ring"))"

 /*****************************************************************/
 /* The webserver's root CA certificate must be saved to a data  */
 /* set so that it may be OPUT to an HFS file for download   @L9A*/
 /*****************************************************************/
 /* If webserver certificate is generated in this exec and       */
 /* it is issued by the self-signed PKI CA, ie. the self-signed  */
 /* PKI CA is the webserver's root CA, export it for OPUT    @L9A*/

 if web_dn ^= "" & signing_ca_label = "" then do          /*@L9A*/
   call logsay2 "Saving the webserver's root CA certificate to a ",
                "data set for OPUT ..."                    /*@L9A*/

   certcmd = "RACDCERT CERTAUTH EXPORT(LABEL('"ca_label"'))",
             " DSN("export_dsn") FORMAT(CERTDER)"          /*@L9A*/
   call tsoserv certcmd
 end
 else                                                       /*@L9A*/
   /* Need to manually export the webserver's root
      CA certificate                                         @L9A*/
   call logsay2 "You need to manually export the webserver's ",
                "root CA certificate."                     /*@L9A*/

 if unix_sec = 0 then do
 /*****************************************************************/
 /* Not setting up z/OS UNIX higher security. However, the       */
 /* daemon does need access to one server service. So, if the    */
 /* daemon user ID is not uid 0, then it must be given read      */
 /* access to FACILITY class profile BPX.SERVER                  */
 /*****************************************************************/
   if strip(daemon_uid,L,'0') ^= "" then do /* if daemon not uid 0 */
     call logsay2 "Giving" daemon "access to BPX.SERVER ..."
     call tsoserv "RDEFINE FACILITY BPX.SERVER"
     call tsoserv "PERMIT BPX.SERVER CLASS(FACILITY)",
         " ID("daemon") ACCESS(READ)"
   end
 end
 else do
   call logsay2 "Setting up or modifying z/OS UNIX security ..."
   if unix_sec = 2 then do
 /*****************************************************************/
 /* Set up z/OS UNIX to operate with a higher level of           */
 /* security than traditional UNIX, by defining BPX.SERVER and   */
 /* BPX.DAEMON classes.                                          */
 /*****************************************************************/
     call tsoserv "RDEFINE FACILITY BPX.SERVER"
     call tsoserv "RDEFINE FACILITY BPX.DAEMON"
     do i = 1 to bpx_userid.0
       call tsoserv "PERMIT BPX.SERVER CLASS(FACILITY)",
         " ID("bpx_userid.i") ACCESS(READ)"
       call tsoserv "PERMIT BPX.DAEMON CLASS(FACILITY)",
         " ID("bpx_userid.i") ACCESS(READ)"
     end
   end


 /*****************************************************************/
 /* To use the higher level of security, you need to establish   */
 /* RACF program control and enable the PKI Services daemon      */
 /* user ID and webserver daemon user ID to access protected     */
 /* UNIX daemon services.                                        */
 /*****************************************************************/
   call tsoserv "PERMIT BPX.SERVER CLASS(FACILITY) ID("daemon")",
    " ACCESS(READ)"
   call tsoserv "PERMIT BPX.DAEMON CLASS(FACILITY) ID("daemon")",
    " ACCESS(READ)"
   call tsoserv "PERMIT BPX.SERVER CLASS(FACILITY) ID("webserver")",
    " ACCESS(UPDATE)"
   call tsoserv "PERMIT BPX.DAEMON CLASS(FACILITY) ID("webserver")",
    " ACCESS(READ)"

   if unix_sec = 2 then do
 /*****************************************************************/
 /* Set the PKI Services daemon and DLLs up for program control  */
 /*****************************************************************/
     call tsoserv "RDEFINE PROGRAM * UACC(NONE)"
     do i = 1 to pgmcntl_dsn.0
      call tsoserv "RALTER PROGRAM * ADDMEM("pgmcntl_dsn.i"//NOPADCHK)",
       " UACC(READ)"
     end
     call tsoserv "SETROPTS WHEN(PROGRAM)"
   end
   call tsoserv "PERMIT * CLASS(PROGRAM)",
    " ID("surrog") ACCESS(READ)"
   call tsoserv "SETROPTS WHEN(PROGRAM) REFRESH"
 end /* unix_sec ^= 0 */

 /***********************************************************************/
 /* Access to the keyring can be controlled either by the profiles in  */
 /* the FACILITY class or by the profile in the RDATALIB class. The set*/
 /* up indicated below uses the FACILITY class. If you want specific   */
 /* access control on the PKI Services keyring and the Web Server      */
 /* keyring, you may use the RDATALIB class. For example, for the      */
 /* daemon's keyring: set up PKISRVD to have CONTROL access on         */
 /* PKISRVD.CAring.LST. For the web server's keyring: set up WEBSRV to */
 /* have READ access on WEBSRV.SSLring.LST.                        @DBA*/
 /*********************************************************************/
 /* Allow the daemon to be a certificate authority and give access to  */
 /* its keyring                                                    @DBC*/
 /*********************************************************************/
 call logsay2 "Allowing the PKI Services daemon to act as a CA ..."
 call tsoserv "RDEFINE FACILITY IRR.DIGTCERT.GENCERT"
 call tsoserv "RDEFINE FACILITY IRR.DIGTCERT.LISTRING"
                                                              /* 1@DBD */
 call tsoserv "PERMIT IRR.DIGTCERT.GENCERT CLASS(FACILITY)",
  " ID("daemon") ACCESS(CONTROL)"
 call tsoserv "PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY)",
  " ID("daemon") ACCESS(READ)"
                                                              /* 2@DBD */

 /******************************************************************/
 /* If the webserver cert and keyring were created in this exec,  */
 /* Allow the webserver to access its keyring                     */
 /******************************************************************/
 if web_dn ^= "" then do                                       /* @D4A*/
   call logsay2 "Allowing the Webserver to access its keyring ..."
   call tsoserv "PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY)",
    " ID("webserver") ACCESS(READ)"
                                                              /* 2@DBD */
 end /* web_dn ^= "" */                                        /* @D4A*/

 /******************************************************************/
 /* Permit the webserver daemon User ID to switch identity to the */
 /* surrogate Id                                                  */
 /******************************************************************/

 call logsay2 "Allowing the Webserver to switch identity to "surrog" ..."
 call tsoserv "SETROPTS CLASSACT(SURROGAT)"
 call tsoserv "RDEFINE SURROGAT BPX.SRV."surrog
 call tsoserv "PERMIT BPX.SRV."surrog" CLASS(SURROGAT)",
  " ID("webserver") ACCESS(READ)"
 call tsoserv "SETROPTS RACLIST(SURROGAT) REFRESH"

                                                               /* @D8D*/
 /**************************************************/
 /* Allow the daemon to use ICSF                   */
 /**************************************************/
 if csfkeys_profile ^= '' | csfserv_profile ^= '' then do      /* @D8M*/
   call logsay2 "Allowing the PKI Services daemon to use ICSF ..."
   call tsoserv "SETROPTS  GENERIC(CSFKEYS CSFSERV)"
   call tsoserv "SETROPTS  GENERIC(CSFKEYS CSFSERV) REFRESH"
                                                               /* @D8D*/
   if csfusers_grp ^= '' then do                               /* @D8M*/
     call tsoserv "CONNECT" daemon "GROUP(" csfusers_grp ")"   /* @D8M*/
     call tsoserv "CONNECT" surrog "GROUP(" csfusers_grp ")"   /* @D8M*/
   end                                                         /* @D8M*/

   if csfkeys_profile ^= '' then do                            /* @02C*/
     call tsoserv "RDEFINE CSFKEYS" csfkeys_profile "UACC(NONE)"
     if csfusers_grp ^= '' then                                /* @D8A*/
       call tsoserv "PERMIT" csfkeys_profile "CLASS(CSFKEYS)", /* @D8A*/
        " ID("csfusers_grp") ACCESS(READ)"                     /* @D8A*/
     else                                                      /* @D8A*/
       call tsoserv "PERMIT" csfkeys_profile "CLASS(CSFKEYS)",
        " ID("daemon") ACCESS(READ)"
     call tsoserv "SETROPTS CLASSACT(CSFKEYS) RACLIST(CSFKEYS)"
     call tsoserv "SETROPTS RACLIST(CSFKEYS) REFRESH"
   end
   if csfserv_profile = 'CSF*' then do                         /* @02C*/
     call tsoserv "RDEFINE CSFSERV" csfserv_profile "UACC(NONE)"
     if csfusers_grp ^= '' then do                             /* @D8A*/
       call tsoserv "PERMIT" csfserv_profile "CLASS(CSFSERV)", /* @D8A*/
        " ID("csfusers_grp") ACCESS(READ)"                     /* @D8A*/
     end                                                       /* @D8A*/
     else do                                                   /* @D8A*/
       call tsoserv "PERMIT" csfserv_profile "CLASS(CSFSERV)",
        " ID("daemon") ACCESS(READ)"
       call tsoserv "PERMIT" csfserv_profile "CLASS(CSFSERV)",
        " ID("surrog") ACCESS(READ)"
     end                                                       /* @D8A*/
     call tsoserv "SETROPTS CLASSACT(CSFSERV) RACLIST(CSFSERV)"
     call tsoserv "SETROPTS RACLIST(CSFSERV) REFRESH"
   end
   else do  /* for bring up ICSF, no key generation               @02A*/
     call tsoserv "RDEFINE CSFSERV" csfserv_profile,
      "UACC(NONE)"                                             /* @02A*/
     call tsoserv "PERMIT" csfserv_profile "CLASS(CSFSERV)",
      " ID("daemon") ACCESS(READ)"                             /* @02A*/
     call tsoserv "SETROPTS CLASSACT(CSFSERV)",
      " RACLIST(CSFSERV)"                                      /* @02A*/
     call tsoserv "SETROPTS RACLIST(CSFSERV) REFRESH"          /* @02A*/
   end                                                         /* @02A*/
 end                                                           /* @D8D*/

 if (key_gen=1) then do                                         /*@L9A*/
 /****************************************************/
 /* Allow the daemon to generate key pairs     @L9A */
 /****************************************************/
   call logsay2 "Allowing the PKI Services daemon to ",
   "generate key pairs ..."                                      /*@L9A*/
   if cryptoz_profile_so ^= '' & cryptoz_profile_user ^= ''
   then do                                                       /*@L9A*/
     call tsoserv "SETROPTS CLASSACT(CRYPTOZ)"                   /*@L9A*/
     call tsoserv "SETROPTS GENERIC(CRYPTOZ)"                    /*@L9A*/
     call tsoserv "SETROPTS RACLIST(CRYPTOZ)"                    /*@L9A*/
     call tsoserv "RDEFINE CRYPTOZ",
          cryptoz_profile_so "UACC(NONE)"                        /*@L9A*/
     call tsoserv "RDEFINE CRYPTOZ",
          cryptoz_profile_user "UACC(NONE)"                      /*@L9A*/
     call tsoserv "PERMIT" cryptoz_profile_so "CLASS(CRYPTOZ)",
          " ID("daemon") ACCESS(UPDATE)"                         /*@D4C*/
     call tsoserv "PERMIT" cryptoz_profile_user "CLASS(CRYPTOZ)",
          " ID("daemon") ACCESS(CONTROL)"                        /*@D4C*/
     call tsoserv "SETROPTS RACLIST(CRYPTOZ) REFRESH"            /*@L9A*/
     call tsoserv "SETROPTS GENERIC(CRYPTOZ) REFRESH"            /*@L9A*/
   end                                                           /*@L9A*/
   if cryptoz_grp ^= '' then do                                  /*@L9A*/
     call tsoserv "CONNECT" daemon "GROUP(" cryptoz_grp ")"      /*@L9A*/
   end                                                           /*@L9A*/
 end

 /****************************************************/
 /* daemon no longer needs access to OCSF           */
 /****************************************************/

 /***********************************************************
  * Tie the daemon user ID to PKI Services started procedure
  ***********************************************************/
  call logsay2 "Creating the STARTED class profile for the daemon ..."
  call tsoserv "RDEFINE STARTED PKISERVD.* STDATA(USER("daemon"))"
  call tsoserv "SETROPTS CLASSACT(STARTED) RACLIST(STARTED)"
  call tsoserv "SETROPTS RACLIST(STARTED) REFRESH"


 /*****************************************************************/
 /* Give the surrogate user ID authority to request certificate  */
 /* generation functions.                                        */
 /*****************************************************************/
 call logsay2 "Allowing "surrog" to request certificate functions ..."
 call tsoserv "SETR GENERIC(FACILITY)"
 /*
  * When a CA Domain is not specified, use the default value for
  * the end user functions generic profile name. When a CA Domain
  * is specified, append a dot/period(".") followed by the CA
  * Domain value to the default value.
  */
 if (ca_domain = "") then                                 /* @L4A*/
    profname = "IRR.RPKISERV.**"                          /* @L4A*/
 else                                                     /* @L4A*/
    profname = "IRR.RPKISERV.*."||ca_domain_trunc         /* @L4A*/

 call tsoserv "RDEFINE FACILITY "||profname               /* @L4C*/
 call tsoserv "PERMIT "||profname||" CLASS(FACILITY)",
              " ID("surrog") ACCESS(CONTROL)"             /* @L4C*/

 /*****************************************************************/
 /* The administrative functions of PKI Services are protected   */
 /* by the IRR.RPKISERV.PKIADMIN FACILITY class resource and     */
 /* optionally by the <domain>.<action>.<template> PKISERV class */
 /* resources.                                                   */
 /* The following commands give UPDATE access on the resource in */
 /* the facility class and READ access on the resources in the   */
 /* PKISERV class to the PKI services group to allow them to act */
 /* on certificate requests and issued certificates.             */
 /*****************************************************************/
 call logsay2 "Creating the profile to protect PKI Admin functions ..."
```

```
/*
 * When a CA Domain is not specified, use the default value for
 * the administrative function profile name. When a CA Domain
 * is specified, append a dot/period(".") followed by the CA
 * Domain value to the default value.
 */
if (ca_domain = "") then                                 /* @L4A*/
   profname = "IRR.RPKISERV.PKIADMIN"                    /* @L4A*/
else                                                     /* @L4A*/
   profname = "IRR.RPKISERV.PKIADMIN."||ca_domain_trunc  /* @L4A*/

call tsoserv "RDEFINE FACILITY "||profname               /* @L4C*/
call tsoserv "PERMIT "||profname||" CLASS(FACILITY)",
             " ID("pkigroup") ACCESS(UPDATE)"            /* @L4C*/
call tsoserv "PERMIT "||profname||" CLASS(FACILITY)",
             " ID("surrog") ACCESS(NONE)"                /* @L4C*/
call tsoserv "SETROPTS RACLIST(FACILITY) REFRESH"

/*************************************************************/
/* If granular control is set, create the corresponding        */
/* profiles for the administrative functions and grant access   */
/* to different groups. The following commands show the        */
/* examples of giving pkigroup authority to perform all the     */
/* functions, except queryreqdetails and querycertdetails which */
/* are permitted for pkigroup2 respectively.                    */
/*************************************************************/
if AdminGranularControl = 1 then do                     /* @LDA */
   call tsoserv "PERMIT "||profname||" CLASS(FACILITY)",
              " ID("pkigroup1") ACCESS(UPDATE)"         /* @DAC */

   call tsoserv "PERMIT "||profname||" CLASS(FACILITY)",
              " ID("pkigroup2") ACCESS(UPDATE)"         /* @DAC */

   call tsoserv "SETROPTS RACLIST(FACILITY) REFRESH"

   call logsay2 "Creating the profiles to granular control",
                " individual PKI Admin function......"  /* @LDA */

   cadomain = ca_domain                                 /* @LDA */
   if(cadomain =="") then                               /* @LDA */
      cadomain = "NOCADOMAIN"                           /* @LDA */

   do r = 1 to template.0                               /* @DAC */

     /* Granular Access to pkigroup1                       @LDA */
     do p = 1 to actions1.0
        call granaccess cadomain template.r pkigroup1 actions1.p
     end                                                /* @DAC */
     call logsay " "                                    /* @LDA */

     /* Granular Access to pkigroup2                       @LDA */
     do q = 1 to actions2.0
        call granaccess cadomain template.r pkigroup2 actions2.q
     end                                                /* @DAC */
     call logsay " "                                    /* @LDA */

   end                                                  /* @LDA */

   /* The actions POSTCERT and CREATECRL are not template based */
   call tsoserv "SETROPTS CLASSACT(PKISERV)"            /* @02A*/
   call tsoserv "SETROPTS RACLIST(PKISERV)"             /* @02A*/
   call tsoserv "RDEFINE PKISERV "|| cadomain || ".",
     ||"POSTCERT"|| " UACC(NONE)"                       /* @DAC */
   call tsoserv "PERMIT "|| cadomain || "." ||"POSTCERT",
     || " CLASS(PKISERV) ID("pkigroup1") ACCESS(READ)"  /* @DAC */

   call tsoserv "RDEFINE PKISERV "|| cadomain || ".",
     ||"CREATECRL"|| " UACC(NONE)"                      /* @DAC */
   call tsoserv "PERMIT "|| cadomain || "." ||"CREATECRL",
     || " CLASS(PKISERV) ID("pkigroup1") ACCESS(READ)"  /* @DAC */

   call tsoserv "SETROPTS RACLIST(PKISERV) REFRESH"     /* @LDA */

end                                                     /* @LDA */

/*****************************/
/* Done. Now write to the log */
/*****************************/
upper daemon vsamhql export_dsn
call logsay " "
call logsay "-----------------------------------------------"
call logsay "Information needed for PKI Services UNIX set up:"
call logsay "-------------------------------------------------"
call logsay " "
call logsay "The daemon user ID is:"
call logsay " " daemon
call logsay " "
call logsay "The VSAM high level qualifier is:"
call logsay " " vsamhlq
call logsay,
"This is needed for the [ObjectStore] section in pkiserv.conf"
call logsay " "
call logsay "The PKI Services' DER encoded certificate is in data set:"
call logsay " " cacert_dsn                                /*@L9C*/
call logsay,
   "If this CA is set up for EST, it must be OPUT to the path specified
   ",
   "by the ESTCAFile keyword in the pkiserv.conf file ",
   "eg /var/pkiserv/estcacert.der ", with the BINARY option"
call logsay " "
if export_dsn ^= "" then do                               /*@L9A*/
  call logsay "The webserver's DER encoded root "
  call logsay "CA certificate is in data set:"
  call logsay " " export_dsn                              /*@L9A*/
  call logsay,
  "This may be OPUT to /var/pkiserv/cacert.der with ",
  "the BINARY option, and distribute it to requestors."
  call logsay " "
end                                                       /*@L9A*/
else do                                                   /*@L9A*/
  call logsay "You need to find the webserver's root CA ",
              "certificate and export it manually."       /*@L9A*/
  call logsay " "                                         /*@L9A*/
end
call logsay "The fully qualified PKI Services' SAF keyring is:"
call logsay " " daemon"/"ca_ring
call logsay,
"This is needed for the [SAF] section in pkiserv.conf"
if ra_label ^= "" then do                                 /*@L4A*/
  call logsay " "                                        /*2@L4A*/
  call logsay "The label of the PKI Services' RA certificate is:"
  call logsay " " ra_label                                /*@L4A*/
  call logsay,                                            /*@L4A*/
  "This is needed for the [SAF] section in pkiserv.conf"  /*@L4A*/
end                                                       /*@L4A*/
call logsay " "
if ca_dn ^= "" then do
  call logsay "The PKI Services CA DN is:"
  call norm_dn ca_dn
  call logsay " " dn
  call logsay "The suffix must match the LDAP suffix in slapd.conf"
end
else
  call logsay "CA certificate not created by this exec"
call logsay " "

if ra_dn ^= "" then do                                     /*@L4A*/
  call logsay "The PKI Services RA DN is:"                 /*@L4A*/
  call norm_dn ra_dn                                       /*@L4A*/
  call logsay " " dn                                      /*2@L4A*/
  call logsay "The suffix must match the LDAP suffix in slapd.conf"
end                                                        /*@L4A*/
else                                                       /*@L4A*/
  call logsay "RA certificate not created by this exec"    /*@L4A*/
call logsay " "                                            /*@L4A*/
call logsay,                                              /*2@L4A*/
"The recommended location for the pkiserv.conf and pkiserv.tmpl is:"
if ca_domain = "" then                                     /*@L4A*/
  call logsay "  /etc/pkiserv"                             /*@L4A*/
else                                                       /*@L4A*/
  call logsay "  /etc/pkiserv/"ca_domain                   /*@L4A*/
call logsay " "                                            /*@L4A*/

call logsay,                                              /*2@L4A*/
"Set the following environment variables in pkiserv.envars:"
if ca_domain = "" then                                     /*@L4A*/
  call logsay "  _PKISERV_CONFIG_PATH=/etc/pkiserv"        /*@L4A*/
else do                                                   /*2@L4A*/
  call logsay "  _PKISERV_CA_DOMAIN="ca_domain             /*@L4A*/
  call logsay "  _PKISERV_CONFIG_PATH=/etc/pkiserv/"ca_domain
end                                                        /*@L4A*/
call logsay " "                                            /*@L4A*/
call logsay,                                              /*2@L4A*/
"Set the following environment variable in your virtual host files:"
```

```
if ca_domain = "" then                                    /*@L4A*/
  call logsay "  _PKISERV_CONFIG_PATH=/etc/pkiserv"        /*@L4A*/
else                                                     /*2@L4A*/
  call logsay "  _PKISERV_CONFIG_PATH_"TRANSLATE(ca_domain),
              "=/etc/pkiserv/"ca_domain                   /*@L4A*/
call logsay " "                                           /*@L4A*/
if web_dn ^= "" then do
  call logsay "The webserver's SAF keyring is:"
  call logsay " " web_ring
  call logsay,
  "This is needed for the KeyFile directive in virtual host files"
  call logsay " "
  call logsay "The Webserver's DN is:"
  call norm_dn web_dn
  call logsay " " dn
  call logsay "The left most RDN must be the webserver's fully",
              "qualified domain name"
end
else
  call logsay,
"Webserver certificate and keyring not created. You must add the CA",
"certificate as a 'trusted root' manually"
call logsay " "
if log_dsn ^= "" then do
  x = OUTTRAP(MSGS.)
  'EXECIO' logdata.0 'DISKW IKYLOGDD (FINIS STEM LOGDATA.'
  'FREE FI(IKYLOGDD)'
  x=OUTTRAP('OFF')
  say "Commands complete. Results written to log data set" log_dsn
end
/***********************/
/* Exit                */
/***********************/
say 'The IKYSETUP EXEC has completed.'
Exit max_return_code

/**********************************************************/
/* tsoserv - echo rc and commands and track highest rc       */
/**********************************************************/
tsoserv:
Parse arg cmd
return_code = 0
skipit= 0
if runopt = "NO" | runopt = "PROMPT" then
  call logsay cmd
if runopt = "PROMPT" then do
  say "Run command (y/n) ?"
  parse pull ans
  if substr(ans,1,1) ^= 'Y' & substr(ans,1,1) ^= 'y' then
    skipit= 1
end
if skipit = 0 then
  if runopt = "YES" | runopt = "PROMPT" then do
    msg_status= MSG('ON')
    x=OUTTRAP('rac_ret.')
    Address TSO cmd
    return_code=rc
    y=OUTTRAP('OFF')
    call logsay 'Return code' return_code 'from->' cmd
    If return_code\=0 then do
      Do j=1 to rac_ret.0
      call logsay rac_ret.j
      end
    end
  end
max_return_code= max(max_return_code,return_code)
return return_code
/* 1@LDD */

/**********************************************************/
/* logsay - echo messages to the terminal and logdata stem   */
/**********************************************************/
logsay:
Parse arg cmd
parse var cmd leftpart " PASSWORD('" pw "') " rightpart
if pw ^= "" then
  cmd= leftpart "PASSWORD('******')" rightpart
say cmd


cmdlen = length(cmd)                               /* @L5A */
Do Ln = 1 to length(cmd) by 252                    /* @L5A */
```

```
  k= logdata.0 + 1
  logdata.k=substr(cmd, Ln, min(252, cmdlen))       /* @L5C */
  logdata.0= k
  cmdlen = cmdlen - min(252, cmdlen)                /* @L5A */
end                                                 /* @L5A */

return 0

/***************************************************************/
/* logsay2 - echo a blank line before echoing the command     */
/***************************************************************/
logsay2:
Parse arg cmd2
call logsay " "
call logsay cmd2

return 0

/***************************************************************/
/* norm_dn - transform the RACF dn keywords to an LDAP dn     */
/***************************************************************/
norm_dn:
parse arg in_dn
parse var in_dn q.1 "('" v.1 "')",
                q.2 "('" v.2 "')",
                q.3 "('" v.3 "')",
                q.4 "('" v.4 "')",
                q.5 "('" v.5 "')",
                q.6 "('" v.6 "')",
                q.7 "('" v.7 "')" rest
dns.= ""
do i = 1 to 7
  q= strip(q.i)
  upper q
  if q = "" then
    leave
  if q = "CN" then
    dns.1= "CN=" ||  v.i
  else
  if q = "T" then
    dns.2= "T=" ||  v.i
  else
  if q = "OU" then
    dns.3= "OU=" ||  v.i
  else
  if q = "O" then
    dns.4= "O=" ||  v.i
  else
  if q = "L" then
    dns.5= "L=" ||  v.i
  else
  if q = "SP" then
    dns.6= "ST=" ||  v.i
  else
    dns.7= "C=" ||  v.i
end
dn= ""
do i = 1 to 7
  if dns.i ^= "" then
    if dn = "" then
      dn= dns.i
    else
      dn= dn || "," || dns.i
end
return 0

/***************************************************************/
/* granaccess - Create profiles and grant access for Granular*/
/* Control                                                     */
/***************************************************************/
granaccess:                                          /* @LDA */
parse arg cadomain template pkigroup action          /* @DAC */

 call tsoserv "RDEFINE PKISERV "|| cadomain || ".",
   ||action || "." || template "UACC(NONE)"         /* @DAC */
 call tsoserv "PERMIT "|| cadomain || "." ||action,
   || "." || template || " CLASS(PKISERV) ID("pkigroup")",
   "ACCESS(READ)"                                    /* @DAC */
return 0                                             /* @LDA */
```

Chapter 31. Other code samples

This topic provides code samples for the following files:

- `httpd.conf`, `vhost80.conf`, `vhost443.conf`, and `vhost1443.conf`, which contain IBM HTTP Server directives. (See "IBM HTTP Server - Powered by Apache configuration directives" on page 649.)
- IKYCDB2, which is a sample to build Db2 objects for the PKI Services object store and issued certificate list (ICL). This JCL is used for `version 0` object store and ICL. (See "IKYCDB2" on page 655.)
- IKYCDBV1, which is a sample to build Db2 objects for the PKI Services object store and issued certificate list (ICL). This JCL is used for `version 1` object store and ICL. (See "IKYCDBV1" on page 658.)
- IKYCVSAM, which is sample IDCAMS JCL to create VSAM data sets (regardless of whether you are using a sysplex or non-sysplex). This JCL is used for `version 0` object store and ICL. (See "IKYCVSAM" on page 661.)
- IKYCVSV1, which is sample IDCAMS JCL to create VSAM data sets (regardless of whether you are using a sysplex or non-sysplex). This JCL is used for `version 1` object store and ICL. (See "IKYCVSV1" on page 665.)
- IKYRVSAM, which is sample IDCAMS JCL to add VSAM record-level sharing (RLS) support. IKYRVSAM reallocates your VSAM data sets in preparation for sharing in a sysplex. This JCL is used for `version 0` object store and ICL. (See "IKYRVSAM" on page 669.)
- IKYRVSV1, which is sample IDCAMS JCL to add VSAM record-level sharing (RLS) support. IKYRVSAM reallocates your VSAM data sets in preparation for sharing in a sysplex. This JCL is used for `version 1` object store and ICL. (See "IKYRVSV1" on page 673.)
- IKYSBIND, which is a sample job to build the Db2 package and plan for the PKI Services object store and issued certificate list (ICL). (See "IKYSBIND" on page 677.)
- IKYSGRNT, which is a sample job to grant execute privilege on the Db2 package for PKI Services to the PKI Services daemon user ID. (See "IKYSGRNT" on page 679.)
- IKYVBKUP, which is sample JCL to back up the PKI Services VSAM data sets using the DFSMSdss DUMP utility. (See "IKYVBKUP" on page 680.)
- IKYVREST, which is sample JCL to restore the PKI Services VSAM data sets from a backup taken with the DFSMSdss DUMP utility. (See "IKYVREST" on page 681.)
- PKISERVD, which is a sample procedure to start PKI Services daemon. (See "PKISERVD sample procedure to start PKI Services daemon" on page 682.)

Note: Other important programs are contained in other chapters:

- IKYSETUP (a REXX exec to set up RACF profiles). See Chapter 30, "The IKYSETUP REXX exec," on page 619.
- `pkiserv.envars` (the PKI Services environment variables file). See "The pkiserv.envars environment variables file" on page 617.
- `pkiserv.conf` (the PKI Services configuration file). See Chapter 28, "The pkiserv.conf configuration file," on page 607.

IBM HTTP Server - Powered by Apache configuration directives

The sample configuration directives for IBM HTTP Server - Powered by Apache are located in the source directory `/usr/lpp/pkiserv/samples/`. The following listings might not be identical to the code samples shipped with the product.

`httpd.conf`

`httpd.conf` is the main configuration file for IBM HTTP Server - Powered by Apache.

```
#--------------------------------------------------------------------#
# Licensed Materials - Property of IBM                               #
# 5650-ZOS                                                           #
# Copyright IBM Corp. 2019                                           #
# Status = HKY77C0                                                   #
#                                                                    #
#--------------------------------------------------------------------#
# Update the ihs-install-dir/conf/httpd.conf file with the following
# directives that might not be present in your httpd.conf file and that
# might be unique to the PKI Services CGI scripts and programs.

#Provides a rule-based rewriting engine to rewrite one URL to another
#URL
LoadModule rewrite_module modules/mod_rewrite.so
#Authenticating with SAF on IBM HTTP Server (z/OS systems)
LoadModule authnz_saf_module modules/mod_authnz_saf.so
LoadModule ibm_ssl_module modules/mod_ibm_ssl.so

# AddType allows you to add to or override the MIME configuration
# file mime.types for specific file types.
# AddType directive is added to the httpd.conf file
# after the TypesConfig Directive
AddType application/x-x509-user-cert .cer
AddType application/x-x509-ca-cert   .der
AddType application/octet-stream     .msi
AddType application/pkix-crl         .crl

#Include other configuration files
#vhost80 - Virtual Host file for non SSL requests
#vhost443 - Virtual Host file for SSL requests with server authentication
#vhost1443 - Virtual Host file for SSL requests with client authentication
Include conf/vhost80.conf
Include conf/vhost443.conf
Include conf/vhost1443.conf
```

vhost80.conf

`vhost80.conf` is the configuration file for non-SSL processing.

```
#--------------------------------------------------------------------#
# Licensed Materials - Property of IBM                               #
# 5650-ZOS                                                           #
# Copyright IBM Corp. 2015, 2019                                     #
# Status = HKY77C0                                                   #
#                                                                    #
# Change-Activity:                                                   #
# $L0=PKIS22S,HKY77A0, 140918, SSD: PKI Services                     #
# $L1=PKIS32B,HKY77B0, 170209, SSD: Browser Currency                 #
# $D1=ZD0090JB,HKY77C0,180817, WFC: Security enhancement             #
#                                                                    #
# Change Descriptions:                                               #
#   A - IBM HTTP Server Support                                      #
#   C - Modified directives to remove support for PKIXEnroll         #
#   C - Remove cacert and PKICEnroll entries                         #
#--------------------------------------------------------------------#
# This Virtual Host file for non SSL requests contains only the
# directives that may be unique to PKI Services. Copy the directives
# needed to the ihs-install-dir/conf/vhost80.conf file if it exists or
# use this file as the base to create one and add the other basic
# directives.

<VirtualHost *:80>
  CharsetOptions      NoTranslateRequestBodies
  AllowEncodedSlashes On

  SetEnv           LIBPATH                              "<application-root>/lib:${LIBPATH}"
  SetEnv           _PKISERV_CONFIG_PATH                 "/etc/pkiserv"

  # Uncomment this section if adding a new CA Domain as described in Chapter 14 - Advance Customization
  # Section - Adding a new CA domain.
  #SetEnv          _PKISERV_CONFIG_PATH_EMPLOYEES       "/etc/pkiserv/employees"
  #SetEnv          _PKISERV_CONFIG_PATH_ADMEMPLOYEES    "/etc/pkiserv/employees"

  RewriteEngine    On
  RewriteRule      ^/(PKIServ|Customers)/ssl-cgi/(.*)         https://<server-domain-name>/$1/ssl-cgi-bin/$2
[R,NE]
  RewriteRule      ^/(PKIServ|Customers)/clientauth-cgi/(.*)  https://<server-domain-name>:1443/$1/clientauth-
cgi-bin/$2 [R,NE]

  # Uncomment this section if adding a new CA Domain as described in Chapter 14 - Advance Customization
  # Section - Adding a new CA domain.
  #RewriteRule     ^/(AdmEmployees|Employees)/ssl-cgi/(.*)         https://<server-domain-name>/$1/ssl-cgi-
bin/$2 [R,NE]
  #RewriteRule     ^/(AdmEmployees|Employees)/clientauth-cgi/(.*)  https://<server-domain-name>:1443/$1/
clientauth-cgi-bin/$2 [R,NE]
```

```
  ScriptAliasMatch   /(PKIServ|Customers)/public-cgi/(.*)     <application-root>/PKIServ/public-cgi/$2

  # Uncomment this line if adding a new CA Domain as described in Chapter 14 - Advance Customization
  # Section - Adding a new CA domain.
  #ScriptAliasMatch  /(AdmEmployees|Employees)/public-cgi/(.*)    <application-root>/PKIServ/public-cgi/$2

  # The User will not be prompted for user id and
  # password and will use the Surrogate User ID
  # PKISERV to access files in this directory
  <Directory <application-root>/PKIServ/public-cgi>
       Options         +Includes
       Require         all granted
       SAFRunAs        PKISERV
  </Directory>

  </VirtualHost>
```

vhost443.conf

`vhost443.conf` is the configuration for server authentication in SSL processing.

```
#-------------------------------------------------------------------#
# Licensed Materials - Property of IBM                              #
# 5650-ZOS                                                          #
# Copyright IBM Corp. 2015, 2019                                    #
# Status = HKY77C0                                                  #
#                                                                   #
# Change-Activity:                                                  #
#  $L0=PKIS22S, HKY77A0, 140918, SSD: PKI Services                  #
#  $L1=PKIS32B, HKY77B0, 170209, SSD: Browser Currency              #
#  $L2=PKIS24E, HKY77C0, 180720, WFC: RCF 7030 and enhanced security  #
#                                                                   #
# Change Descriptions:                                              #
#   A - IBM HTTP Server Support                                     #
#   C - Modified directives to remove support for PKIXEnroll        #
#   A - Add Enroll over Secure Transport (EST) CGI alias, remove http #
#       RewriteRule                                                 #
#-------------------------------------------------------------------#
# This Virtual Host file for SSL requests with server authentication
# contains only the directives that may be unique to PKI Services.
# Copy the directives needed to the ihs-install-dir/conf/vhost443.conf
# file if it exists or use this file as the base to create one and add
# the other basic directives.
Listen 443
<VirtualHost *:443>
  SetEnv           LIBPATH                               "<application-root>/lib:${LIBPATH}"
  SetEnv           _PKISERV_CONFIG_PATH                  "/etc/pkiserv"

  #Environment variables for tracing EST
  #SetEnv          _PKISERV_EST_TRACE                    0xff
  #SetEnv          _PKISERV_EST_TRACE_FILE               /tmp/pkiest.%.trc

  # Uncomment this section if adding a new CA Domain as described in Chapter 14 - Advance Customization,
  # Section - Adding a new CA domain.
  #SetEnv          _PKISERV_CONFIG_PATH_EMPLOYEES        "/etc/pkiserv/employees"
  #SetEnv          _PKISERV_CONFIG_PATH_ADMEMPLOYEES     "/etc/pkiserv/employees"

  Keyfile          /saf SSLring
  SSLEnable
  SSLClientAuth    None

  RewriteEngine    On
  RewriteRule      ^/(PKIServ|Customers)/ssl-cgi/(.*)         https://<server-domain-name>/$1/ssl-cgi-bin/$2
[R,NE]
  RewriteRule      ^/(PKIServ|Customers)/clientauth-cgi/(.*)  https://<server-domain-name>:1443/$1/clientauth-
cgi-bin/$2 [R,NE,L]

  # Uncomment this section if adding a new CA Domain as described in Chapter 14 - Advance Customization
  # Section - Adding a new CA domain.
  #RewriteRule     ^/(AdmEmployees|Employees)/ssl-cgi/(.*)         https://<server-domain-name>/$1/ssl-cgi-
bin/$2 [R,NE]
  #RewriteRule     ^/(AdmEmployees|Employees)/clientauth-cgi/(.*)  https://<server-domain-name>:1443/$1/
clientauth-cgi-bin/$2 [R,NE,L]

  ScriptAliasMatch ^/(PKIServ|Customers)/(public-cgi|ssl-cgi-bin)/(.*)          "<application-root>/
PKIServ/$2/$3"
  ScriptAliasMatch ^/.well-known/est(.*)                                        "<application-root>/PKIServ/
ssl-cgi-bin/pkiest"

  # Uncomment this line if adding a new CA Domain as described in Chapter 14 - Advance Customization
  # Section - Adding a new CA domain.
  #ScriptAliasMatch ^/(AdmEmployees|Employees)/(public-cgi|ssl-cgi-bin)/(.*)     "<application-root>/
PKIServ/$2/$3"
  #ScriptAliasMatch ^/.well-known/est/employees/(.*)                             "<application-root>/PKIServ/
ssl-cgi-bin/pkiest"

  AliasMatch       /PKIServ/cacerts/(.*)                 /var/pkiserv/$1
  <Directory   /var/pkiserv>
      AuthName          PublicUser
      Require           all granted
      SAFRunAs          PKISERV
  </Directory>

  AliasMatch       /PKIServ/PKICEnroll/(.*)               <application-root>/ActiveX/PKICEnroll/$1
  <Directory   <application-root>/ActiveX/PKICEnroll>
      AuthName          PublicUser
      Require           all granted
      SAFRunAs          PKISERV
  </Directory>

  # The User will be prompted to enter a RACF User ID
  # and password and will use the same RACF User ID
  # and password to access files in this directory
  <Directory <application-root>/PKIServ/ssl-cgi-bin/auth>
      AuthName          AuthenticatedUser
      AuthType          Basic
      AuthBasicProvider saf
      Require           valid-user
      SAFRunAs          %%CLIENT%%
  </Directory>

  # The User will be prompted to enter a RACF User ID
  # and password but will use the Surrogate User ID
  # PKISERV to access files in this directory
  <Directory <application-root>/PKIServ/ssl-cgi-bin/surrogateauth>
      AuthName          SAFSurrogateUser
      AuthType          Basic
      AuthBasicProvider saf
      Require           valid-user
      SAFRunAs          PKISERV
  </Directory>

  # The User will not be prompted for user id and
  # password and will use the Surrogate User ID
  # PKISERV to access files in this directory
  <Directory <application-root>/PKIServ/ssl-cgi-bin>
      AuthName          SurrogateUser
      AuthType          None
      Require           all granted
      SAFRunAs          PKISERV
  </Directory>

  <LocationMatch "^/(PKIServ|Customers)/ssl-cgi-bin(/(auth|surrogateauth))?/cagetcert.rexx">
      Charsetoptions TranslateAllMimeTypes
  </LocationMatch>

  # Uncomment this section if adding a new CA Domain as described in Chapter 14 - Advance Customization
  # Section - Adding a new CA domain.
  #<LocationMatch "^/(AdmEmployees|Employees)/ssl-cgi-bin(/(auth|surrogateauth))?/cagetcert.rexx">
  #    Charsetoptions TranslateAllMimeTypes
  #</LocationMatch>

</VirtualHost>
```

vhost1443.conf

`vhost1443.conf` is the configuration file for client authentication in SSL processing.

```
#--------------------------------------------------------------------#
# Licensed Materials - Property of IBM                               #
# 5650-ZOS                                                           #
# Copyright IBM Corp. 2015,2019                                      #
# Status = HKY77C0                                                   #
#                                                                    #
# Change-Activity:                                                   #
#  $L0=PKIS22S, HKY77A0, 140918, SSD: PKI Services                   #
#  $L1=PKIS32B, HKY77B0, 170209, SSD: Browser Currency               #
#  $L2=PKIS24E, HKY77C0, 180823, WFC: RFC 7030 and enhanced security  #
#                                                                    #
# Change Descriptions:                                               #
#   A - IBM HTTP Server Support                                      #
#   C - Modified directives to remove support for PKIXEnroll         #
#   A - Add Enroll over Secure Transport (EST) CGI alias, remove http #
#       RewriteRule                                                  #
#--------------------------------------------------------------------#
# This Virtual Host file for SSL requests with client authentication
# contains only the directives that may be unique to PKI Services. Copy
# the directives needed to the ihs-install-dir/conf/vhost1443.conf file
# if it exists or use this file as the base to create one and add the
# other basic directives.
Listen 1443
<VirtualHost *:1443>
  SetEnv           LIBPATH                          "<application-root>/lib:${LIBPATH}"
  SetEnv           _PKISERV_CONFIG_PATH             "/etc/pkiserv"
  # Environment variables for tracing CMP and EST
  #SetEnv          _PKISERV_CMP_TRACE                     0xff
  #SetEnv          _PKISERV_CMP_TRACE_FILE                /tmp/pkicmp.%.trc
  #SetEnv          _PKISERV_EST_TRACE                     0xff
  #SetEnv          _PKISERV_EST_TRACE_FILE                /tmp/pkiest.%.trc

  # Uncomment this section if adding a new CA Domain as described in Chapter 14 - Advance Customization
  # Section - Adding a new CA domain.
  #SetEnv          _PKISERV_CONFIG_PATH_EMPLOYEES        "/etc/pkiserv/employees"
  #SetEnv          _PKISERV_CONFIG_PATH_ADMEMPLOYEES     "/etc/pkiserv/employees"

  Keyfile           /saf SSLring
  SSLEnable
  SSLClientAuth     Required

  RewriteEngine     on
  RewriteRule       ^/(PKIServ|Customers)/ssl-cgi/(.*)     https://<server-domain-name>/$1/ssl-cgi-bin/$2
[R,NE,L]

  # Uncomment this section if adding a new CA Domain as described in Chapter 14 - Advance Customization
  # Section - Adding a new CA domain.
  #RewriteRule       ^/(AdmEmployees|Employees)/ssl-cgi/(.*)     https://<server-domain-name>/$1/ssl-cgi-
bin/$2 [R,NE,L]

  ScriptAliasMatch  ^/(PKIServ|Customers)/(clientauth-cgi|clientauth-cgi-bin)/(.*)           "<application-
root>/PKIServ/clientauth-cgi-bin/$3"

  ScriptAliasMatch ^/.well-known/est/(.*)   "<application-root>/PKIServ/clientauth-cgi-bin/pkiest"

  # Uncomment this line if adding a new CA Domain as described in Chapter 14 - Advance Customization
  # Section - Adding a new CA domain.
  #ScriptAliasMatch  ^/(AdmEmployees|Employees)/(clientauth-cgi|clientauth-cgi-bin)/(.*)     "<application-
root>/PKIServ/clientauth-cgi-bin/$3"

  #Uncomment this line if adding a new CA domain for EST.
  #ScriptAliasMatch ^/.well-known/est/employees/(.*)   "<application-root>/PKIServ/clientauth-cgi-bin/pkiest"

  AliasMatch        /PKIServ/PKICEnroll/(.*)         <application-root>/ActiveX/PKICEnroll/$1
  <Directory   <application-root>/ActiveX/PKICEnroll>
      AuthName          PublicUser
      Require           all granted
      SAFRunAs          PKISERV
  </Directory>

  # The User will not be prompted to enter a RACF User ID
  # and password but will use the Surrogate User ID
  # PKISERV to access files in this directory
  <Directory        <application-root>/PKIServ/clientauth-cgi-bin>
      AuthName          RenewRevokeUser
      AuthType          Basic
      AuthBasicProvider saf
      Require           all granted
      SAFRunAs          PKISERV
  </Directory>

  # The User will be prompted for a client certificate
  # for SSL authentication.
  <Directory        <application-root>/PKIServ/clientauth-cgi-bin/auth>
      AuthName          AuthenticatedAdmin
      AuthType          Basic
      AuthBasicProvider saf
      Require           valid-user
      SAFRunAs          %%CERTIF_REQ%%
  </Directory>

  <LocationMatch    "^/(PKIServ|Customers)/clientauth-cgi-bin/auth/pkicmp">
      CharsetOptions  NoTranslateRequestBodies
  </LocationMatch>

  # Uncomment this line if adding a new CA Domain as described in Chapter 14 - Advance Customization
  # Section - Adding a new CA domain.
  #<LocationMatch    "^/(AdmEmployees|Employees)/clientauth-cgi-bin/auth/pkicmp">
  #    CharsetOptions  NoTranslateRequestBodies
  #</LocationMatch>

</VirtualHost>
```

WebSphere Liberty

`server.xml`

`server.xml` is the main server configuration file for Liberty.

```
#-------------------------------------------------------------------#
# Licensed Material - Property of IBM                               #
# 5650-ZOS                                                          #
# Copyright IBM Corp. 2017                                          #
# Status = HKY77B0                                                  #
#-------------------------------------------------------------------#

<server decsription="new server">
    <!-- Enable features -->
    <featureManager>
       <feature>jsp-2.2</feature>
       <feature>localConnector-1.0</feature>
       <feature>ssl-1.0</feature>
       <feature>appSecurity-2.0</feature>
       <feature>servlet-3.0</feature>
       <feature>ejbLite-3.1</feature>
       <feature>zosSecurity-1.0</feature>
    <featureManager>

    <httpEndpoint host="pkiserver.mycompany.com"
     httpPort="9080" httpsPort="9440" id="defaultHttpEndpoint">
     <sslOptions sslRef="ServerAuthSSLConfig" />
    </httpEndpoint>

    <applicationMonitor updateTrigger="mbean" />

    <ssl id="ServerAuthSSLConfig"
           trustStoreRef="defaultKeyStore"
           keyStoreRef="defaultKeyStore"
           clientAuthentication="false" />

    <ssl id="ClientAuthConfig"
           keyStoreRef="defaultKeyStore"
           trustStoreRef="defaultKeyStore"
           clientAuthentication="true"
           sslProtocol="TLS" />

    <keyStore id="defaultKeyStore"
location="safkeyring://libsvr/LibertyKeyring"
     type="JCERACFKS" password="password" fileBased="false"
readOnly="true" />

    <safRegistry id="saf" />

<!-- Uncomment the following line if you wish to use saf
authorization with delegation -->
<!-- <safAuthorization enableDelegation="true" /> -->

    <syncToOSThread appEnabled="true" />

    <library id="global">
      <fileset dir="/usr/lpp/pkiserv/lib" includes="*.jar"
               scanInterval="5s" />
      <fileset dir="/usr/lpp/pkiserv/lib64" includes="*.so"
               scanInterval="5s" />
    </library>

    <enterpriseApplication id="PKI"
    location="/etc/pki/pkiear/PKIServ_root.ear"
    name="PKI">
      <application-bnd>
        <security-role name="PKIAdmin">
           <user name="admin1" />
        </security-role>
        <security-role name="SAFuser">
           <special-subject type="ALL_AUTHENTICATED_USERS" />
        </security-role>
        <security-role name="PKISurrogate">
           <user name="PKISERV" />
           <run-as userid="PKISERV" />
        </security-role>
      </application-bnd>
      <classloader commonLibraryRef="global" />
    </enterpriseApplication>

    <enterpriseApplication id="PKI2"
    location="/etc/pki/pkiear/PKIServ_subca.ear"
    name="PKI2">
      <application-bnd>
```

```
         <security-role name="PKIAdmin">
            <user name="admin2" />
         </security-role>
         <security-role name="SAFuser">
            <special-subject type="ALL_AUTHENTICATED_USERS" />
         </security-role>
         <security-role name="PKISurrogate">
            <user name="PKISERV" />
            <run-as userid="PKISERV" />
         </security-role>
       </application-bnd>
       <classloader commonLibraryRef="global" />
     </enterpriseApplication>

</server>
```

IKYCDB2

IKYCDB2 is a sample to create Db2 objects for the object store and issued certificate list (ICL). IKYCDB2 is a member of SYS1.SAMPLIB.

Note: The following listing might not be identical to the code sample shipped with the product. For the most current sample, see SYS1.SAMPLIB member IKYCDB2.

```
--**********************************************************************
--* SAMPLE: IKYCDB2
--*
--* Licensed Materials - Property of IBM
--* 5650-ZOS
--* Copyright IBM Corp.  2011, 2013
--* Status = HKY7790
--*
--**********************************************************************
--
-- This sample may be used to create the DB2 database using SPUFI that
-- PKI Services utilizes to store certificate requests and issued
-- certificates.
--
--**********************************************************************
--
-- Before using this sample, you may need to make the following
-- modifications:
--
-- 1) Change all the occurences of 'SYSDEFLT' to the storage group you
--    want to contain the PKI Services DB2 tablespaces if SYSDEFLT
--    if SYSDEFLT is not suitable for your installation.
--
-- 2) Change all the occurences of 'MASTERCA' to the package name.
--    The package name should match the first eight characters of
--    the CA domain name.
--
-- 3) Change all the occurences of 'IKYPKIDB' to the database name of
--    your choosing. If you are running multiple PKI CA domains, each
--    domain must have a unique database name.
--
-- You will also need certain DB2 privileges to use this sample.
-- These privileges are indicated in the comments preceding each
-- set of SQL instructions.
--
-- If you have already run this sample before, uncomment these
-- statements to drop existing indexes, tables, and tablespaces.
-- In order to DROP these objects, the user must have at least one
-- of the following privileges:
--      - Ownership of the indices, tables and tablespaces
--      - DBADM authority on the IKYPKIDB database
--      - SYSADM or SYSCTRL authority
--
--**********************************************************************
--
-- DROP INDEX MASTERCA.TIDAIX;
-- DROP INDEX MASTERCA.OREQAIX;
-- DROP INDEX MASTERCA.OSTATAIX;
-- DROP TABLE MASTERCA.OST;
--
-- DROP INDEX MASTERCA.IREQAIX;
-- DROP INDEX MASTERCA.ISTATAIX;
-- DROP TABLE MASTERCA.ICL;
--
-- DROP TABLESPACE IKYPKIDB.OSTSPACE;
-- DROP TABLESPACE IKYPKIDB.ICLSPACE;
--
--*********************************************************************
--
-- If you have already run this sample before, uncomment these
-- statements to drop existing database.  In order to DROP the
-- database, the user must have at least one of the following
-- privileges:
--     - DROP privilege on the IKYPKIDB database
--     - DBADM or DBCTRL authority on the IKYPKIDB database
--     - SYSADM or SYSCTRL authority
--
--*********************************************************************
--
-- DROP DATABASE IKYPKIDB;
--
-- COMMIT;
--
--*********************************************************************
--
-- The following statement creates the PKI Services DB2 database.
-- To create the database, the user must have at least one of
-- the following privileges:
--     - CREATEDBA privilege
--     - CREATEDBC privilege
--     - SYSADM or SYSCTRL authority
--
--*********************************************************************
CREATE DATABASE IKYPKIDB STOGROUP SYSDEFLT;
-
--*********************************************************************
--
-- The following statements create the tablespaces used for the
-- PKI Services ObjectStore and ICL tables.  To create these
-- tablespaces, the user must have at least one of the following
-- privileges:
--     - CREATETSA privilege on the IKYPKIDB database
--     - DBADM, DBCTRL, or DBMAINT authority for the
--       IKYPKIDB database
--     - SYSADM or SYSCTRL authority
--
--*********************************************************************
CREATE TABLESPACE OSTSPACE IN IKYPKIDB
     LOCKSIZE       ROW
     SEGSIZE        4
     PCTFREE        0
     BUFFERPOOL     BP32K
     USING STOGROUP SYSDEFLT
     PRIQTY         144400
;
CREATE TABLESPACE ICLSPACE IN IKYPKIDB
     LOCKSIZE       ROW
     SEGSIZE        4
     PCTFREE        0
     BUFFERPOOL     BP32K
     USING STOGROUP SYSDEFLT
     PRIQTY         144400
;
--*********************************************************************
--
-- The following statement creates the ObjectStore table.
-- To create the table, the user must have at least one of the
-- following privileges:
--     - CREATETAB privilege on the IKYPKIDB database
--     - DBADM, DBCTRL, or DBMAINT authority for the
--       IKYPKIDB database
--     - SYSADM or SYSCTRL authority
--
-- The table name MUST be <package name>.OST
--
--*********************************************************************
CREATE TABLE MASTERCA.OST(
      RECORD_KEY                  BINARY(4)        NOT NULL,
      RECORD_STATE                BINARY(4)        NOT NULL,
      REQDATA_LEN                 INTEGER          NOT NULL,
      REQUESTOR_NAME              VARCHAR(32)              ,
      TRANS_ID                    CHAR(24)         NOT NULL,
      COMMENT                     VARCHAR(64)              ,
      ISSUED_TIME                 TIMESTAMP        NOT NULL,
      LAST_CHANGE_TIME              TIMESTAMP         NOT NULL,
      TEMPLATE_NICKNAME             VARCHAR(8)                ,
      SERIAL_NUM                    BINARY(4)                 ,
      REQDATA                       VARBINARY(32512)  NOT NULL,
      PRIMARY KEY (RECORD_KEY)
      )
 IN IKYPKIDB.OSTSPACE
 ;
--*********************************************************************
--
-- The following statement creates the indices for the
-- ObjectStore table.  To create these indices, the user must
-- have at least one of the following privileges:
--      - INDEX privilege on the ObjectStore table
--      - Ownership of the ObjectStore table
--      - DBADM authority for the IKYPKIDB database
--      - SYSADM or SYSCTRL authority
--
--*********************************************************************
--
-- The following statement creates the ICL table.  To create the
-- table, the user must have at least one of the following
-- privileges:
--      - CREATETAB privilege on the IKYPKIDB database
--      - DBADM, DBCTRL, or DBMAINT authority for the
--        IKYPKIDB database
--      - SYSADM or SYSCTRL authority
--
-- The table name MUST be <package name>.ICL
--
--*********************************************************************
CREATE TABLE MASTERCA.ICL(
      SERIAL_NUM                    BINARY(4)         NOT NULL,
      CERT_STATE                    BINARY(4)         NOT NULL,
      CERT_LEN                      INTEGER           NOT NULL,
      REQUESTOR_NAME                VARCHAR(32)       NOT NULL,
      REVOKE_DATE                   TIMESTAMP                 ,
      INVALID_DATE                  TIMESTAMP                 ,
      REVOKE_REASON                 INTEGER                   ,
      COMMENT                       VARCHAR(64)               ,
      ISSUED_TIME                   TIMESTAMP         NOT NULL,
      LAST_CHANGE_TIME              TIMESTAMP         NOT NULL,
      TEMPLATE_NICKNAME             VARCHAR(8)                ,
      OBFUS_PW                      VARBINARY(33)             ,
      PROCESS_FLAGS                 BINARY(4)                 ,
      KEYID                         BINARY(20)                ,
      CRLDP_NUM                     INTEGER                   ,
      EXPIRE_EPOCH_DAYS             INTEGER           NOT NULL,
      EXPIRE_DATE                   TIMESTAMP         NOT NULL,
      KU_DIGTSIG                    BINARY(1)                 ,
      KU_NONRPU                     BINARY(1)                 ,
      KU_KEYENC                     BINARY(1)                 ,
      KU_DATAENC                    BINARY(1)                 ,
      KU_KEYAGR                     BINARY(1)                 ,
      KU_CRTSGN                     BINARY(1)                 ,
      KU_CRLSGN                     BINARY(1)                 ,
      KU_ENCONLY                    BINARY(1)                 ,
      KU_DECONLY                    BINARY(1)                 ,
      EKU_SEVAUTH                   BINARY(1)                 ,
      EKU_CLIAUTH                   BINARY(1)                 ,
      EKU_CODESGN                   BINARY(1)                 ,
      EKU_EMLPROT                   BINARY(1)                 ,
      EKU_TMESTMP                   BINARY(1)                 ,
      EKU_OCSPSGN                   BINARY(1)                 ,
      EKU_MSSCLNON                  BINARY(1)                 ,
      PREV_SERIAL_NUM               BINARY(4)                 ,
      SUBJ_DN                       VARCHAR(1024)     NOT NULL,
      X509CERT                      VARBINARY(10240)  NOT NULL,
      PRIMARY KEY (SERIAL_NUM)
      )
 IN IKYPKIDB.ICLSPACE
 ;
--
--*********************************************************************
--
-- The following statement creates the indices for the ICL
-- table.  To create these indices, the user must have at
-- least one of the following privileges:
--      - INDEX privilege on the ICL table
--      - Ownership of the ICL table
--      - DBADM authority for the IKYPKIDB database
--      - SYSADM or SYSCTRL authority
```

```
--
--*********************************************************************
CREATE UNIQUE INDEX MASTERCA.SERIX    ON MASTERCA.ICL (SERIAL_NUM);
CREATE        INDEX MASTERCA.IREQAIX  ON MASTERCA.ICL (REQUESTOR_NAME);
CREATE        INDEX MASTERCA.ISTATAIX ON MASTERCA.ICL (CERT_STATE,
                                                       CERT_LEN,
                                                       REQUESTOR_NAME);
COMMIT;
```

IKYCDBV1

IKYCDBV1 is a sample to create Db2 objects for the object store and issued certificate list (ICL). IKYCDBV1 is a member of SYS1.SAMPLIB.

Note: The following listing might not be identical to the code sample shipped with the product. For the most current sample, see SYS1.SAMPLIB member IKYCDBV1.

```
--*********************************************************************
--* SAMPLE: IKYCDBV1
--*
--* Licensed Materials - Property of IBM
--* 5650-ZOS
--* Copyright IBM Corp. 2017
--* Status = HKY77B0
--*
--*********************************************************************
--
-- This sample may be used to create the DB2 database using SPUFI that
-- PKI Services utilizes to store certificate requests and issued
-- certificates.
--
-- The DB2 database created by this sample supports improved tracking
-- of SCEP requests and certificates issued for those requests that is
-- available in PKI Services V2R3 (HKY77B0) and later releases.  For the
-- sample that creates the original DB2 database format utilized by PKI
-- Services V2R2 (HKY77A0) and earlier releases, please see the IKYCDB2
-- sample.
--
--*********************************************************************
--
-- Before using this sample, you may need to make the following
-- modifications:
--
-- 1) Change all the occurences of 'SYSDEFLT' to the storage group you
--    want to contain the PKI Services DB2 tablespaces if SYSDEFLT
--    if SYSDEFLT is not suitable for your installation.
--
-- 2) Change all the occurences of 'MASTERC1' to the package name.
--    In choosing this name, it is recommended to use the first
--    seven characters of the associated CA domain name, followed
--    by a character that will uniquely identify this package name
--    with the DBRM assigned to this package.  This sample assumes
--    that 'MASTERC1' is the assocaited CA Domain name, and the
--    last character '1' is used to associate this package with the
--    IKYPDBR1 DBRM.  The resulting name used here must match the
--    value used in the IKYSBIND job.
--
-- 3) Change all the occurences of 'IKYPKID1' to the database name of
--    your choosing. If you are running multiple PKI CA domains, each
--    domain must have a unique database name.  If you have existing
--    databases for these domains from a prior version of PKI
--    Services, you must choose a different name.
--
-- You will also need certain DB2 privileges to use this sample.
-- These privileges are indicated in the comments preceding each
-- set of SQL instructions.
--
--*********************************************************************
--
-- If you have already run this sample before, uncomment these
-- statements to drop existing indexes, tables, and tablespaces.
-- In order to DROP these objects, the user must have at least one
-- of the following privileges:
--     - Ownership of the indices, tables and tablespaces
--     - DBADM authority on the IKYPKID1 database
--     - SYSADM or SYSCTRL authority
```

```
--
--*********************************************************************
--
-- DROP INDEX MASTERC1.TIDAIX;
-- DROP INDEX MASTERC1.OREQAIX;
-- DROP INDEX MASTERC1.OSCEPAIX;
-- DROP INDEX MASTERC1.OSTATAIX;
-- DROP TABLE MASTERC1.OSTV1;
--
-- DROP INDEX MASTERC1.IREQAIX;
-- DROP INDEX MASTERC1.ISCEPAIX;
-- DROP INDEX MASTERC1.ISTATAIX;
-- DROP TABLE MASTERC1.ICLV1;
--
-- DROP TABLESPACE IKYPKID1.OSTSPACE;
-- DROP TABLESPACE IKYPKID1.ICLSPACE;
--
--*********************************************************************
--
-- If you have already run this sample before, uncomment these
-- statements to drop existing database.  In order to DROP the
-- database, the user must have at least one of the following
-- privileges:
--     - DROP privilege on the IKYPKID1 database
--     - DBADM or DBCTRL authority on the IKYPKID1 database
--     - SYSADM or SYSCTRL authority
--
--*********************************************************************
--
-- DROP DATABASE IKYPKID1;
--
-- COMMIT;
--
--*********************************************************************
--
-- The following statement creates the PKI Services DB2 database.
-- To create the database, the user must have at least one of
-- the following privileges:
--     - CREATEDBA privilege
--     - CREATEDBC privilege
--     - SYSADM or SYSCTRL authority
--
--*********************************************************************
--
CREATE DATABASE IKYPKID1 STOGROUP SYSDEFLT;
--
--*********************************************************************
--
-- The following statements create the tablespaces used for the
-- PKI Services ObjectStore and ICL tables.  To create these
-- tablespaces, the user must have at least one of the following
-- privileges:
--     - CREATETSA privilege on the IKYPKID1 database
--     - DBADM, DBCTRL, or DBMAINT authority for the
--       IKYPKID1 database
--     - SYSADM or SYSCTRL authority
--
--*********************************************************************
CREATE TABLESPACE OSTSPACE IN IKYPKID1
     LOCKSIZE       ROW
     SEGSIZE        4
     PCTFREE        0
     BUFFERPOOL     BP32K
     USING STOGROUP SYSDEFLT
     PRIQTY         144400
;
CREATE TABLESPACE ICLSPACE IN IKYPKID1
     LOCKSIZE       ROW
     SEGSIZE        4
     PCTFREE        0
     BUFFERPOOL     BP32K
     USING STOGROUP SYSDEFLT
     PRIQTY         144400
;
--*********************************************************************
--
-- The following statement creates the ObjectStore table.
-- To create the table, the user must have at least one of the
-- following privileges:
--     - CREATETAB privilege on the IKYPKID1 database
--     - DBADM, DBCTRL, or DBMAINT authority for the
--       IKYPKID1 database
--     - SYSADM or SYSCTRL authority
--
-- The table name MUST be <package name>.OSTV1
--
--*********************************************************************
CREATE TABLE MASTERC1.OSTV1(
      RECORD_KEY                  BINARY(4)         NOT NULL,
      RECORD_STATE                BINARY(4)         NOT NULL,
      REQDATA_LEN                 INTEGER           NOT NULL,
      REQUESTOR_NAME              VARCHAR(32)               ,
      TRANS_ID                    CHAR(24)          NOT NULL,
      COMMENT                     VARCHAR(64)               ,
      ISSUED_TIME                 TIMESTAMP         NOT NULL,
      LAST_CHANGE_TIME            TIMESTAMP         NOT NULL,
      TEMPLATE_NICKNAME           VARCHAR(8)                ,
      SERIAL_NUM                  BINARY(4)                 ,
      SCEP_TRANSID                VARCHAR(128)      NOT NULL,
      DB_VERSION                  BINARY(1)         NOT NULL,
      REQDATA                     VARBINARY(32380)  NOT NULL,
      PRIMARY KEY (RECORD_KEY)
      )
 IN IKYPKID1.OSTSPACE
 ;
--*********************************************************************
--
-- The following statement creates the indices for the
-- ObjectStore table.  To create these indices, the user must
-- have at least one of the following privileges:
--     - INDEX privilege on the ObjectStore table
--     - Ownership of the ObjectStore table
--     - DBADM authority for the IKYPKID1 database
--     - SYSADM or SYSCTRL authority
--
--*********************************************************************
 CREATE UNIQUE INDEX MASTERC1.RKEYIX
              ON    MASTERC1.OSTV1 (RECORD_KEY);
 CREATE       INDEX MASTERC1.TIDAIX
              ON    MASTERC1.OSTV1 (TRANS_ID);
 CREATE       INDEX MASTERC1.OSCEPAIX
              ON    MASTERC1.OSTV1 (SCEP_TRANSID);
 CREATE       INDEX MASTERC1.OREQAIX
              ON    MASTERC1.OSTV1 (REQUESTOR_NAME);
 CREATE       INDEX MASTERC1.OSTATAIX
              ON    MASTERC1.OSTV1 (RECORD_STATE  ,
                                    REQDATA_LEN   ,
                                    REQUESTOR_NAME);
--*********************************************************************
--
-- The following statement creates the ICL table.  To create the
-- table, the user must have at least one of the following
-- privileges:
--     - CREATETAB privilege on the IKYPKID1 database
--     - DBADM, DBCTRL, or DBMAINT authority for the
--       IKYPKID1 database
--     - SYSADM or SYSCTRL authority
--
-- The table name MUST be <package name>.ICLV1
--
--*********************************************************************
CREATE TABLE MASTERC1.ICLV1(
      SERIAL_NUM                  BINARY(4)         NOT NULL,
      CERT_STATE                  BINARY(4)         NOT NULL,
      CERT_LEN                    INTEGER           NOT NULL,
      REQUESTOR_NAME              VARCHAR(32)       NOT NULL,
      REVOKE_DATE                 TIMESTAMP                 ,
      INVALID_DATE                TIMESTAMP                 ,
      REVOKE_REASON               INTEGER                   ,
      COMMENT                     VARCHAR(64)               ,
      ISSUED_TIME                 TIMESTAMP         NOT NULL,
      LAST_CHANGE_TIME            TIMESTAMP         NOT NULL,
      TEMPLATE_NICKNAME           VARCHAR(8)                ,
      OBFUS_PW                    VARBINARY(33)             ,
      PROCESS_FLAGS               BINARY(4)                 ,
      KEYID                       BINARY(20)                ,
      CRLDP_NUM                   INTEGER                   ,
      EXPIRE_EPOCH_DAYS           INTEGER           NOT NULL,
      EXPIRE_DATE                 TIMESTAMP         NOT NULL,
      KU_DIGTSIG                  BINARY(1)                 ,
      KU_NONRPU                   BINARY(1)                 ,
      KU_KEYENC                   BINARY(1)                 ,
      KU_DATAENC                  BINARY(1)                 ,
      KU_KEYAGR                   BINARY(1)                 ,
      KU_CRTSGN                     BINARY(1)                 ,
      KU_CRLSGN                     BINARY(1)                 ,
      KU_ENCONLY                    BINARY(1)                 ,
      KU_DECONLY                    BINARY(1)                 ,
      EKU_SEVAUTH                   BINARY(1)                 ,
      EKU_CLIAUTH                   BINARY(1)                 ,
      EKU_CODESGN                   BINARY(1)                 ,
      EKU_EMLPROT                   BINARY(1)                 ,
      EKU_TMESTMP                   BINARY(1)                 ,
      EKU_OCSPSGN                   BINARY(1)                 ,
      EKU_MSSCLNON                  BINARY(1)                 ,
      PREV_SERIAL_NUM               BINARY(4)                 ,
      SUBJ_DN                       VARCHAR(1024)     NOT NULL,
      SCEP_TRANSID                  VARCHAR(128)      NOT NULL,
      DB_VERSION                    BINARY(1)         NOT NULL,
      X509CERT                      VARBINARY(10240)  NOT NULL,
      PRIMARY KEY (SERIAL_NUM)
      )
 IN IKYPKID1.ICLSPACE
 ;
--**********************************************************************
--
-- The following statement creates the indices for the ICL
-- table.  To create these indices, the user must have at
-- least one of the following privileges:
--     - INDEX privilege on the ICL table
--     - Ownership of the ICL table
--     - DBADM authority for the IKYPKID1 database
--     - SYSADM or SYSCTRL authority
--
--**********************************************************************
CREATE UNIQUE INDEX MASTERC1.SERIX
              ON    MASTERC1.ICLV1 (SERIAL_NUM);
CREATE        INDEX MASTERC1.ISCEPAIX
              ON    MASTERC1.ICLV1 (SCEP_TRANSID);
CREATE        INDEX MASTERC1.IREQAIX
              ON    MASTERC1.ICLV1 (REQUESTOR_NAME);
CREATE        INDEX MASTERC1.ISTATAIX
              ON    MASTERC1.ICLV1 (CERT_STATE    ,
                                    CERT_LEN      ,
                                    REQUESTOR_NAME);
COMMIT;
```

IKYCVSAM

IKYCVSAM contains sample IDCAMS JCL to create VSAM data sets. IKYCVSAM is installed as a member of SYS1.SAMPLIB.

Use IKYCVSAM if you are creating VSAM data sets for the first time, regardless of whether you intend to use Parallel Sysplex support. However, if you intend to use Parallel Sysplex support, execute the IKYRVSAM job *after* this job to add VSAM record-level sharing (RLS) support. (See "IKYRVSAM" on page 669.)

Note: The following listing might not be identical to the code sample shipped with the product. For the most current sample, see SYS1.SAMPLIB member IKYCVSAM.

```
//IKYCVSAM JOB <job card parameters>
//*********************************************************************
//*  SAMP:      IKYCVSAM                                               *
//*                                                                    *
//*    Licensed Materials - Property of IBM                            *
//*    5694-A01                                                        *
//*    Copyright IBM Corp. 2001, 2011                                  *
//*    Status = HKY7780                                                *
//*                                                                    *
//*********************************************************************
//*                                                                    *
//*  This sample JCL may be used to create the VSAM data sets          *
//*  PKI Services utilizes to store certificate requests and           *
//*  issued certificates.                                              *
//*                                                                    *
```

```
//*******************************************************************
//*                                                                 *
//*  Caution: This is neither a JCL procedure nor a complete job.   *
//*  Before using this job step, you will have to make the following *
//*  modifications:                                                 *
//*                                                                 *
//*  1) Change the job card to meet your system requirements.       *
//*                                                                 *
//*  2) If you wish to change the data set qualifiers from the      *
//*     default value change all occurrences of "PKISRVD.VSAM"      *
//*     to a preferred value. If you choose to modify this value, be *
//*     be sure to also modify the sample configuration file        *
//*     appropriately(/etc/pkiserv/pkiserv.conf). If you are using  *
//*     multiple CA Domains, IBM recommends using the first eight   *
//*     characters of the CA Domain as one of the data set          *
//*     qualifiers.                                                 *
//*                                                                 *
//*  3) If you are using VSAM record level sharing (RLS), perform   *
//*     the following steps:                                        *
//*                                                                 *
//*     a) Replace the VOL(vvvvvv) statements in the DEFKSDS step   *
//*        with STORCLAS(class-name) where class-name is the name of *
//*        the storage class defined for VSAM RLS.                  *
//*                                                                 *
//*     b) Remove the VOL(vvvvvv) statements from the DEFALTDX step. *
//*                                                                 *
//*     c) Remove all the SPANNED and CISIZE statements.            *
//*                                                                 *
//*     If not using VSAM RLS, change all occurrences of vvvvvv to  *
//*     the VOLSER value appropriate for the system this job is to be *
//*     run on. Do not remove the SPANNED and CISIZE statements.    *
//*                                                                 *
//*                                                                 *
//*  4) If you wish to change the default userid to own the VSAM    *
//*     data set, change the OWNER(PKISRVD) operand to the userid you *
//*     want to own the data sets. If you choose to modify this value *
//*     ensure you have modified the sample setup REXX exec (IKYSETUP)*
//*     to account for this change.                                 *
//*                                                                 *
//*  5) If you wish to change either the primary or secondary record *
//*     allocation sizes for either the OST or ICL datasets from the *
//*     default value, update the RECORDS(50 50) operands on the    *
//*     DEFINE CLUSTER or DEFINE ALTERNATE INDEX commands.          *
//*                                                                 *
//*  **Note, do not change any of the numeric values other than     *
//*       CYL or TRK                                                *
//*-----------------------------------------------------------------*
//*
//*-----------------------------------------------------------------*
//* Delete existing clusters, paths, alt indexes
//*-----------------------------------------------------------------*
//DELCLUST EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DELETE -
        PKISRVD.VSAM.OST -
        CLUSTER -
        PURGE -
        ERASE
    DELETE -
        PKISRVD.VSAM.ICL -
        CLUSTER -
        PURGE -
        ERASE
    IF MAXCC LT 9 THEN SET MAXCC = 0
/*
//*-----------------------------------------------------------------*
//* Define KSDS                                                     *
//*-----------------------------------------------------------------*
//DEFKSDS  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
```

```
    DEFINE CLUSTER -
(NAME(PKISRVD.VSAM.OST) -
        VOL(vvvvvv) -
        RECSZ(1024 32756) -
        INDEXED -
        NOREUSE -
        KEYS(4 0) -
        SHR(2) -
        CYL(3,1) -
        LOG(NONE) -
        OWNER(PKISRVD) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.DA) -
        CISZ(4096) -
        SPANNED) -
      INDEX -
        (NAME(PKISRVD.VSAM.OST.IX))

    DEFINE CLUSTER -
        (NAME(PKISRVD.VSAM.ICL) -
        VOL(vvvvvv) -
        RECSZ(1024 32756) -
        INDEXED -
        NOREUSE -
        KEYS(4 0) -
        SHR(2) -
        CYL(3,1) -
        LOG(NONE) -
        OWNER(PKISRVD) ) -
      DATA -
        (NAME(PKISRVD.VSAM.ICL.DA) -
        CISZ(4096) -
        SPANNED) -
      INDEX -
        (NAME(PKISRVD.VSAM.ICL.IX))
/*
//*-------------------------------------------------------------------*
//* Repro record of all binary zeros into KSDS                        *
//*-------------------------------------------------------------------*
//MKZEROS  EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   *

//SYSUT2   DD   DSN=&&GENTMP,UNIT=SYSDA,DISP=(,PASS),
//         DCB=(RECFM=FB,LRECL=80,BLKSIZE=640),SPACE=(TRK,(1,1))
//SYSIN    DD   *
 GENERATE MAXFLDS=4,MAXLITS=80
 RECORD FIELD=(20,X'0000000000000000000000000000000000000000',,1),
        FIELD=(20,X'0000000000000000000000000000000000000000',,21),
        FIELD=(20,X'0000000000000000000000000000000000000000',,41),
        FIELD=(20,X'0000000000000000000000000000000000000000',,61)
/*
//REPROKSD EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSDATA  DD   DSN=*.MKZEROS.SYSUT2,DISP=(OLD,DELETE)
//SYSIN    DD   *
   REPRO INFILE(SYSDATA) -
      OUTDATASET(PKISRVD.VSAM.OST)
   REPRO INFILE(SYSDATA) -
      OUTDATASET(PKISRVD.VSAM.ICL)
/*
//*-------------------------------------------------------------------*
//* Define ALTERNATE INDEX and PATH                                   *
//*-------------------------------------------------------------------*
//DEFALTDX EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.OST.AIX) -
        RELATE(PKISRVD.VSAM.OST)-
        VOL(vvvvvv) -
        TRK(5,1) -
```

```
         KEYS(24 44) ) -
       DATA -
         (NAME(PKISRVD.VSAM.OST.AIX.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.OST.AIX.IX))
     DEFINE PATH -
         (NAME(PKISRVD.VSAM.OST.PATH) -
          PATHENTRY(PKISRVD.VSAM.OST.AIX))
     DEFINE ALTERNATEINDEX -
        (NAME(PKISRVD.VSAM.OST.STATAIX) -
         RELATE(PKISRVD.VSAM.OST)-
         VOL(vvvvvv) -
         TRK(5,1) -
         KEYS(40 4) ) -
       DATA -
         (NAME(PKISRVD.VSAM.OST.STATAIX.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.OST.STATAIX.IX))
     DEFINE PATH -
         (NAME(PKISRVD.VSAM.OST.STATUS) -
          PATHENTRY(PKISRVD.VSAM.OST.STATAIX))
     DEFINE ALTERNATEINDEX -
        (NAME(PKISRVD.VSAM.ICL.STATAIX) -
         RELATE(PKISRVD.VSAM.ICL)-
         VOL(vvvvvv) -
         TRK(5,1) -
         KEYS(40 4) ) -
       DATA -
         (NAME(PKISRVD.VSAM.ICL.STATAIX.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.ICL.STATAIX.IX))
     DEFINE PATH -
         (NAME(PKISRVD.VSAM.ICL.STATUS) -
          PATHENTRY(PKISRVD.VSAM.ICL.STATAIX))
     DEFINE ALTERNATEINDEX -
        (NAME(PKISRVD.VSAM.OST.REQAIX) -
         RELATE(PKISRVD.VSAM.OST)-
         VOL(vvvvvv) -
         TRK(5,1) -
         KEYS(32 12) ) -
       DATA -
         (NAME(PKISRVD.VSAM.OST.REQAIX.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.OST.REQAIX.IX))
     DEFINE PATH -
         (NAME(PKISRVD.VSAM.OST.REQUESTR) -
          PATHENTRY(PKISRVD.VSAM.OST.REQAIX))
     DEFINE ALTERNATEINDEX -
        (NAME(PKISRVD.VSAM.ICL.REQAIX) -
         RELATE(PKISRVD.VSAM.ICL)-
         VOL(vvvvvv) -
         TRK(5,1) -
         KEYS(32 12) ) -
       DATA -
         (NAME(PKISRVD.VSAM.ICL.REQAIX.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.ICL.REQAIX.IX))
     DEFINE PATH -
         (NAME(PKISRVD.VSAM.ICL.REQUESTR) -
          PATHENTRY(PKISRVD.VSAM.ICL.REQAIX))
/*
//*------------------------------------------------------------------*
//* BUILD ALTERNATE INDEX                                            *
//*------------------------------------------------------------------*
//BLDINDEX EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.AIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.STATAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.ICL) -
```

```
         OUTDATASET(PKISRVD.VSAM.ICL.STATAIX)
     BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
         OUTDATASET(PKISRVD.VSAM.OST.REQAIX)
     BLDINDEX INDATASET(PKISRVD.VSAM.ICL) -
         OUTDATASET(PKISRVD.VSAM.ICL.REQAIX)
/*
//*--------------------------------------------------------------------*
//* Print out the cluster                                              *
//*--------------------------------------------------------------------*
//PRTCLUST EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    PRINT -
        INDATASET(PKISRVD.VSAM.OST) CHAR
    PRINT -
        INDATASET(PKISRVD.VSAM.ICL) CHAR
/*
```

IKYCVSV1

IKYCVSV1 contains sample IDCAMS JCL to create VSAM data sets. IKYCVSV1 is installed as a member of SYS1.SAMPLIB.

Use IKYCVSV1 if you are creating VSAM data sets for the first time, regardless of whether you intend to use Parallel Sysplex support. However, if you intend to use Parallel Sysplex support, execute the IKYRVSV1 job *after* this job to add VSAM record-level sharing (RLS) support. (See "IKYRVSV1" on page 673.)

Note: The following listing might not be identical to the code sample shipped with the product. For the most current sample, see SYS1.SAMPLIB member IKYCVSV1.

```
//IKYCVSV1 JOB <job card parameters>
//*********************************************************************
//*  SAMP:      IKYCVSV1                                              *
//*                                                                   *
//*    Licensed Materials - Property of IBM                           *
//*    5694-A01                                                       *
//*    Copyright IBM Corp. 2017                                       *
//*    Status = HKY77B0                                               *
//*                                                                   *
//*********************************************************************
//*                                                                   *
//*  This sample JCL may be used to create the VSAM data sets         *
//*  PKI Services utilizes to store certificate requests and          *
//*  issued certificates.                                             *
//*                                                                   *
//*  The VSAM data sets created by this sample support improved       *
//*  tracking of SCEP requests and certificates issued for those      *
//*  requests that is available in PKI Services V2R3 (HKY77B0) and    *
//*  later releases.  For the sample that creates the original VSAM   *
//*  data set format utilized by PKI Services V2R2 (HKY77A0) and      *
//*  earlier releases, please see the IKYCVSAM sample.
*
//*                                                                   *
//*********************************************************************
//*                                                                   *
//*  Caution: This is neither a JCL procedure nor a complete job.     *
//*  Before using this job step, you will have to make the following  *
//*  modifications:                                                   *
//*                                                                   *
//*  1) Change the job card to meet your system requirements.         *
//*                                                                   *
//*  2) If you wish to change the data set qualifiers from the        *
//*     default value change all occurrences of "PKISRVD.VSAM"        *
//*     to a preferred value. If you choose to modify this value, be  *
//*     be sure to also modify the sample configuration file          *
//*     appropriately(/etc/pkiserv/pkiserv.conf). If you are using    *
//*     multiple CA Domains, IBM recommends using the first eight     *
//*     characters of the CA Domain as one of the data set            *
//*     qualifiers.                                                   *
//*                                                                   *
//*  3) If you are using VSAM record level sharing (RLS), perform     *
//*     the following steps:                                          *
//*                                                                   *
//*     a) Replace the VOL(vvvvvv) statements in the DEFKSDS step     *
//*        with STORCLAS(class-name) where class-name is the name of  *
//*        the storage class defined for VSAM RLS.                    *
//*                                                                   *
//*     b) Remove the VOL(vvvvvv) statements from the DEFALTDX step.  *
//*                                                                   *
//*     c) Remove all the SPANNED and CISIZE statements.              *
//*                                                                   *
//*     If not using VSAM RLS, change all occurrences of vvvvvv to    *
//*     the VOLSER value appropriate for the system this job is to be *
//*     run on. Do not remove the SPANNED and CISIZE statements.      *
//*                                                                   *
//*                                                                   *
//*  4) If you wish to change the default userid to own the VSAM      *
//*     data set, change the OWNER(PKISRVD) operand to the userid you *
//*     want to own the data sets. If you choose to modify this value *
//*     ensure you have modified the sample setup REXX exec (IKYSETUP)*
//*     to account for this change.                                   *
//*                                                                   *
//*  5) If you wish to change either the primary or secondary record  *
//*     allocation sizes for either the OST or ICL datasets from the  *
//*     default value, update the RECORDS(50 50) operands on the      *
//*     DEFINE CLUSTER or DEFINE ALTERNATE INDEX commands.            *
//*                                                                   *
//*  **Note, do not change any of the numeric values other than       *
//*       CYL or TRK                                                  *
//*                                                                   *
//*-------------------------------------------------------------------*
//*
//*-------------------------------------------------------------------*
//* Delete existing clusters, paths, alt indexes
//*-------------------------------------------------------------------*
//DELCLUST EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DELETE -
        PKISRVD.VSAM.OST -
        CLUSTER -
        PURGE -
        ERASE
    DELETE -
        PKISRVD.VSAM.ICL -
        CLUSTER -
        PURGE -
        ERASE
    IF MAXCC LT 9 THEN SET MAXCC = 0
/*
//*-------------------------------------------------------------------*
//* Define KSDS                                                       *
//*-------------------------------------------------------------------*
//DEFKSDS  EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DEFINE CLUSTER -
(NAME(PKISRVD.VSAM.OST) -
        VOL(vvvvvv) -
        RECSZ(1024 32756) -
        INDEXED -
        NOREUSE -
        KEYS(4 0) -
        SHR(2) -
        CYL(3,1) -
        LOG(NONE) -
        OWNER(PKISRVD) ) -
```

```
      DATA -
        (NAME(PKISRVD.VSAM.OST.DA) -
        CISZ(4096) -
        SPANNED) -
      INDEX -
        (NAME(PKISRVD.VSAM.OST.IX))

    DEFINE CLUSTER -
        (NAME(PKISRVD.VSAM.ICL) -
        VOL(vvvvvv) -
        RECSZ(1024 32756) -
        INDEXED -
        NOREUSE -
        KEYS(4 0) -
        SHR(2) -
        CYL(3,1) -
        LOG(NONE) -
        OWNER(PKISRVD) ) -
      DATA -
        (NAME(PKISRVD.VSAM.ICL.DA) -
        CISZ(4096) -
        SPANNED) -
      INDEX -
        (NAME(PKISRVD.VSAM.ICL.IX))
/*
//*-------------------------------------------------------------------*
//* Repro record of all binary zeros into KSDS                        *
//*-------------------------------------------------------------------*
//MKZEROS  EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   *

//SYSUT2   DD   DSN=&&GENTMP,UNIT=SYSDA,DISP=(,PASS),
//         DCB=(RECFM=FB,LRECL=512,BLKSIZE=0),SPACE=(TRK,(1,1))
//SYSIN    DD   *
 GENERATE MAXFLDS=22,MAXLITS=512
 RECORD FIELD=(20,X'0000000000000000000000000000000000000000',,1),
        FIELD=(20,X'0000000000000000000000000000000000000000',,21),
        FIELD=(20,X'0000000000000000000000000000000000000000',,41),
        FIELD=(20,X'0000000000000000000000000000000000000000',,61),
        FIELD=(20,X'0000000000000000000000000000000000000000',,81),
        FIELD=(20,X'0000000000000000000000000000000000000000',,101),
        FIELD=(20,X'0000000000000000000000000000000000000000',,121),
        FIELD=(20,X'0000000000000000000000000000000000000000',,161),
        FIELD=(20,X'0000000000000000000000000000000000000000',,181),
        FIELD=(20,X'0000000000000000000000000000000000000000',,201),
        FIELD=(20,X'0000000000000000000000000000000000000000',,221),
        FIELD=(20,X'0000000000000000000000000000000000000000',,261),
        FIELD=(20,X'0000000000000000000000000000000000000000',,281),
        FIELD=(20,X'0000000000000000000000000000000000000000',,301),
        FIELD=(20,X'0000000000000000000000000000000000000000',,321),
        FIELD=(20,X'0000000000000000000000000000000000000000',,361),
        FIELD=(20,X'0000000000000000000000000000000000000000',,381),
        FIELD=(20,X'0000000000000000000000000000000000000000',,401),
        FIELD=(20,X'0000000000000000000000000000000000000000',,421),
        FIELD=(20,X'0000000000000000000000000000000000000000',,461),
        FIELD=(20,X'0000000000000000000000000000000000000000',,481),
        FIELD=(12,X'000000000000000000000000',,501),
/*
//REPROKSD EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSDATA  DD   DSN=*.MKZEROS.SYSUT2,DISP=(OLD,DELETE)
//SYSIN    DD   *
   REPRO INFILE(SYSDATA) -
      OUTDATASET(PKISRVD.VSAM.OST)
   REPRO INFILE(SYSDATA) -
      OUTDATASET(PKISRVD.VSAM.ICL)
/*
//*-------------------------------------------------------------------*
//* Define ALTERNATE INDEX and PATH                                   *
//*-------------------------------------------------------------------*
//DEFALTDX EXEC PGM=IDCAMS
```

```
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.OST.AIX) -
        RELATE(PKISRVD.VSAM.OST)-
        VOL(vvvvvv) -
        TRK(5,1) -
        KEYS(24 44) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.AIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.OST.AIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.OST.PATH) -
         PATHENTRY(PKISRVD.VSAM.OST.AIX))
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.OST.STATAIX) -
        RELATE(PKISRVD.VSAM.OST)-
        VOL(vvvvvv) -
        TRK(5,1) -
        KEYS(40 4) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.STATAIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.OST.STATAIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.OST.STATUS) -
         PATHENTRY(PKISRVD.VSAM.OST.STATAIX))
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.ICL.STATAIX) -
        RELATE(PKISRVD.VSAM.ICL)-
        VOL(vvvvvv) -
        TRK(5,1) -
        KEYS(40 4) ) -
      DATA -
        (NAME(PKISRVD.VSAM.ICL.STATAIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.ICL.STATAIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.ICL.STATUS) -
         PATHENTRY(PKISRVD.VSAM.ICL.STATAIX))
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.OST.REQAIX) -
        RELATE(PKISRVD.VSAM.OST)-
        VOL(vvvvvv) -
        TRK(5,1) -
        KEYS(32 12) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.REQAIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.OST.REQAIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.OST.REQUESTR) -
         PATHENTRY(PKISRVD.VSAM.OST.REQAIX))
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.ICL.REQAIX) -
        RELATE(PKISRVD.VSAM.ICL)-
        VOL(vvvvvv) -
        TRK(5,1) -
        KEYS(32 12) ) -
      DATA -
        (NAME(PKISRVD.VSAM.ICL.REQAIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.ICL.REQAIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.ICL.REQUESTR) -
         PATHENTRY(PKISRVD.VSAM.ICL.REQAIX))
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.OST.SCEPAIX) -
        RELATE(PKISRVD.VSAM.OST)-
        VOL(vvvvvv) -
        TRK(5,1) -
```

```
         KEYS(128 241) ) -
       DATA -
         (NAME(PKISRVD.VSAM.OST.SCEPAIX.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.OST.SCEPAIX.IX))
     DEFINE PATH -
         (NAME(PKISRVD.VSAM.OST.SCEPTID) -
          PATHENTRY(PKISRVD.VSAM.OST.SCEPAIX))
     DEFINE ALTERNATEINDEX -
        (NAME(PKISRVD.VSAM.ICL.SCEPQAIX) -
         RELATE(PKISRVD.VSAM.ICL)-
         VOL(vvvvvv) -
         TRK(5,1) -
         KEYS(128 241) ) -
       DATA -
         (NAME(PKISRVD.VSAM.ICL.SCEPAIX.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.ICL.SCEPAIX.IX))
     DEFINE PATH -
         (NAME(PKISRVD.VSAM.ICL.SCEPTID) -
          PATHENTRY(PKISRVD.VSAM.ICL.SCEPAIX))
/*
//*------------------------------------------------------------------*
//* BUILD ALTERNATE INDEX                                            *
//*------------------------------------------------------------------*
//BLDINDEX EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.AIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.STATAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.ICL) -
        OUTDATASET(PKISRVD.VSAM.ICL.STATAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.REQAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.ICL) -
        OUTDATASET(PKISRVD.VSAM.ICL.REQAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.SCEPAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.ICL) -
        OUTDATASET(PKISRVD.VSAM.ICL.SCEPAIX)
/*
//*------------------------------------------------------------------*
//* Print out the cluster                                            *
//*------------------------------------------------------------------*
//PRTCLUST EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
   PRINT -
       INDATASET(PKISRVD.VSAM.OST) CHAR
   PRINT -
       INDATASET(PKISRVD.VSAM.ICL) CHAR
/*
```

IKYRVSAM

IKYRVSAM contains sample IDCAMS JCL to migrate the PKI Services VSAM data sets to support VSAM record-level sharing (RLS) when you intend to use Parallel Sysplex support. IKYRVSAM is installed as a member of SYS1.SAMPLIB.

Execute this job *after* executing the IKYCVSAM job. This job renames the VSAM data sets created by IKYCVSAM and copies their contents to newly allocated RLS data sets.

Note: The following listing might not be identical to the code sample shipped with the product. For the most current sample, see SYS1.SAMPLIB member IKYRVSAM.

```
//IKYRVSAM JOB <job card parameters>
//*********************************************************************
//*  SAMP:      IKYRVSAM                                              *
//*                                                                   *
//*    Licensed Materials - Property of IBM                           *
//*    5650-ZOS                                                       *
//*    Copyright IBM Corp. 2002, 2017                                 *
//*    Status = HKY77B0                                               *
//*                                                                   *
//*********************************************************************
//*                                                                   *
//*  This sample JCL may be used to reallocate the VSAM data sets     *
//*  in a storage class acceptable to VSAM record level sharing (RLS).*
//*  This is a prerequisite to using PKI Services SYSPLEX support.    *
//*                                                                   *
//*  The VSAM data sets reallocated by this sample utilize the        *
//*  original dats set format employed by PKI Services V2R2 (HKY77A0) *
//*  and earlier releases.  For the sample that creates the revised   *
//*  format for improved tracking of SCEP requests and the            *
//*  certificates issued for those requests that is available in      *
//*  PKI Services V2R3 (HKY77B0) and later releases, please see the   *
//*  IKYRVSV1 sample.                                                 *
//*                                                                   *
//*********************************************************************
//*                                                                   *
//*  Caution: This is neither a JCL procedure nor a complete job.     *
//*  Before using this job step, you will have to make the following  *
//*  modifications:                                                   *
//*                                                                   *
//*  1) Change the job card to meet your system requirements.         *
//*                                                                   *
//*  2) Change the STORCLAS statements to provide the name of the     *
//*     storage class defined for use with VSAM RLS.                  *
//*                                                                   *
//*  3) This job assumes you are using the default VSAM data set      *
//*     names (all have high level qualifiers "PKISRVD.VSAM"). If     *
//*     you have changed these data set names, you will need to       *
//*     modify the source data set names in the ALTER                 *
//*     statements of the RENAMEDS step. If you are using             *
//*     multiple CA Domains, IBM recommends using the first eight     *
//*     characters of the CA Domain as one of the data set            *
//*     qualifiers.                                                   *
//*                                                                   *
//*  4) This job creates destination data sets with the same default  *
//*     names as the source data sets. (The source data sets are      *
//*     renamed.) If you wish to use different destination data set   *
//*     names, you will need to modify the data set names in all      *
//*     steps except the RENAMEDS step. If you modify these names,    *
//*     be sure to also modify your configuration file                *
//*     appropriately(/etc/pkiserv/pkiserv.conf). If you are using    *
//*     multiple CA Domains, IBM recommends using the first eight     *
//*     characters of the CA Domain as one of the data set            *
//*     qualifiers.                                                   *
//*                                                                   *
//*  5) This job renames the source data sets to begin with high      *
//*     level qualifiers "PKISRVD.OLDVSAM". If you wish to change     *
//*     these names, you will need to do so in the RENAMEDS and       *
//*     and REPROCL steps. If you are using multiple CA Domains,      *
//*     IBM recommends using the first eight characters of the CA     *
//*     Domain as one of the data set qualifiers.                     *
//*                                                                   *
//*  6) If you wish to change either the primary or secondary space   *
//*     allocation sizes for either the OST or ICL datasets from the  *
//*     default value, update the CYL or TRK operands on the          *
//*     DEFINE CLUSTER or DEFINE ALTERNATE INDEX commands.            *
//*                                                                   *
//*  **Note, do not change any of the numeric values other than       *
//*        CYL or TRK                                                 *
//*-------------------------------------------------------------------*
```

```
//*
//*-----------------------------------------------------------------*
//* Rename source clusters, alternate indexes and PATH              *
//*-----------------------------------------------------------------*
//RENAMEDS  EXEC PGM=IDCAMS
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
    ALTER -
        PKISRVD.VSAM.OST -
        NEWNAME(PKISRVD.OLDVSAM.OST)
    ALTER -
        PKISRVD.VSAM.OST.* -
        NEWNAME(PKISRVD.OLDVSAM.OST.*)
    ALTER -
        PKISRVD.VSAM.ICL -
        NEWNAME(PKISRVD.OLDVSAM.ICL)
    ALTER -
        PKISRVD.VSAM.ICL.* -
        NEWNAME(PKISRVD.OLDVSAM.ICL.*)
    ALTER -
        PKISRVD.VSAM.OST.AIX.IX -
        NEWNAME(PKISRVD.OLDVSAM.OST.AIX.IX)
    IF LASTCC EQ 8 THEN -
      DO
        SET MAXCC EQ 0
        ALTER -
            PKISRVD.VSAM.AIX.IX -
            NEWNAME(PKISRVD.OLDVSAM.OST.AIX.IX)
      END
    ALTER -
        PKISRVD.VSAM.OST.AIX.DA -
        NEWNAME(PKISRVD.OLDVSAM.OST.AIX.DA)
    IF LASTCC EQ 8 THEN -
      DO
        SET MAXCC EQ 0
        ALTER -
            PKISRVD.VSAM.AIX.DA -
            NEWNAME(PKISRVD.OLDVSAM.OST.AIX.DA)
      END
    ALTER -
        PKISRVD.VSAM.OST.STATAIX.* -
        NEWNAME(PKISRVD.OLDVSAM.OST.STATAIX.*)
    ALTER -
        PKISRVD.VSAM.OST.REQAIX.* -
        NEWNAME(PKISRVD.OLDVSAM.OST.REQAIX.*)
    ALTER -
        PKISRVD.VSAM.ICL.STATAIX.* -
        NEWNAME(PKISRVD.OLDVSAM.ICL.STATAIX.*)
    ALTER -
        PKISRVD.VSAM.ICL.REQAIX.* -
        NEWNAME(PKISRVD.OLDVSAM.ICL.REQAIX.*)
/*
//*-----------------------------------------------------------------*
//* Define destination Clusters                                     *
//*-----------------------------------------------------------------*
//DEFKSDS  EXEC PGM=IDCAMS,COND=(8,LE)
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DEFINE CLUSTER -
        (NAME(PKISRVD.VSAM.OST) -
        STORCLAS(class-name) -
        RECSZ(1024 32756) -
        INDEXED -
        NOREUSE -
        KEYS(4 0) -
        SHR(2) -
        CYL(3,1) -
        LOG(NONE) -
        OWNER(PKISRVD) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.DA)) -
      INDEX -
```

```
         (NAME(PKISRVD.VSAM.OST.IX))

     DEFINE CLUSTER -
         (NAME(PKISRVD.VSAM.ICL) -
         STORCLAS(class-name) -
         RECSZ(1024 32756) -
         INDEXED -
         NOREUSE -
         KEYS(4 0) -
         SHR(2) -
         LOG(NONE) -
         CYL(3,1) -
         OWNER(PKISRVD) ) -
       DATA -
         (NAME(PKISRVD.VSAM.ICL.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.ICL.IX))
/*
//*------------------------------------------------------------------*
//* Repro source cluster to destination cluster                      *
//*------------------------------------------------------------------*
//REPROCL EXEC PGM=IDCAMS,COND=(8,LE)
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
   REPRO INDATASET(PKISRVD.OLDVSAM.OST) -
      OUTDATASET(PKISRVD.VSAM.OST)
   REPRO INDATASET(PKISRVD.OLDVSAM.ICL) -
      OUTDATASET(PKISRVD.VSAM.ICL)
/*
//*------------------------------------------------------------------*
//* Define ALTERNATE INDEX AND PATH                                  *
//*------------------------------------------------------------------*
//DEFALTDX EXEC PGM=IDCAMS,COND=(8,LE)
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.OST.AIX) -
        RELATE(PKISRVD.VSAM.OST)-
        TRK(5,1) -
        KEYS(24 44) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.AIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.OST.AIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.OST.PATH) -
         PATHENTRY(PKISRVD.VSAM.OST.AIX))
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.OST.STATAIX) -
        RELATE(PKISRVD.VSAM.OST)-
        TRK(5,1) -
        KEYS(40 4) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.STATAIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.OST.STATAIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.OST.STATUS) -
         PATHENTRY(PKISRVD.VSAM.OST.STATAIX))
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.ICL.STATAIX) -
        RELATE(PKISRVD.VSAM.ICL)-
        TRK(5,1) -
        KEYS(40 4) ) -
      DATA -
        (NAME(PKISRVD.VSAM.ICL.STATAIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.ICL.STATAIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.ICL.STATUS) -
         PATHENTRY(PKISRVD.VSAM.ICL.STATAIX))
    DEFINE ALTERNATEINDEX -
```

```
        (NAME(PKISRVD.VSAM.OST.REQAIX) -
         RELATE(PKISRVD.VSAM.OST)-
         TRK(5,1) -
         KEYS(32 12) ) -
       DATA -
         (NAME(PKISRVD.VSAM.OST.REQAIX.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.OST.REQAIX.IX))
     DEFINE PATH -
         (NAME(PKISRVD.VSAM.OST.REQUESTR) -
          PATHENTRY(PKISRVD.VSAM.OST.REQAIX))
     DEFINE ALTERNATEINDEX -
        (NAME(PKISRVD.VSAM.ICL.REQAIX) -
         RELATE(PKISRVD.VSAM.ICL)-
         TRK(5,1) -
         KEYS(32 12) ) -
       DATA -
         (NAME(PKISRVD.VSAM.ICL.REQAIX.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.ICL.REQAIX.IX))
     DEFINE PATH -
         (NAME(PKISRVD.VSAM.ICL.REQUESTR) -
          PATHENTRY(PKISRVD.VSAM.ICL.REQAIX))
/*
//*-----------------------------------------------------------------*
//* BUILD ALTERNATE INDEX                                           *
//*-----------------------------------------------------------------*
//BLDINDEX EXEC PGM=IDCAMS,COND=(8,LE)
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.AIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.STATAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.ICL) -
        OUTDATASET(PKISRVD.VSAM.ICL.STATAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.REQAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.ICL) -
        OUTDATASET(PKISRVD.VSAM.ICL.REQAIX)
/*
```

IKYRVSV1

IKYRVSV1 contains sample IDCAMS JCL to migrate the PKI Services VSAM data sets to support VSAM record-level sharing (RLS) when you intend to use Parallel Sysplex support. IKYRVSV1 is installed as a member of SYS1.SAMPLIB.

Execute this job *after* executing the IKYCVSV1 job. This job renames the VSAM data sets created by IKYCVSV1 and copies their contents to newly allocated RLS data sets. (See "IKYCVSV1" on page 665.)

Note: The following listing might not be identical to the code sample shipped with the product. For the most current sample, see SYS1.SAMPLIB member IKYRVSV1.

```
//IKYRVSV1 JOB <job card parameters>
//*******************************************************************
//*  SAMP:      IKYRVSV1                                            *
//*                                                                 *
//*    Licensed Materials - Property of IBM                         *
//*    5650-ZOS                                                     *
//*    Copyright IBM Corp. 2017                                     *
//*    Status = HKY77B0                                             *
//*                                                                 *
//*******************************************************************
//*                                                                 *
//*  This sample JCL may be used to reallocate the VSAM data sets     *
//*  in a storage class acceptable to VSAM record level sharing (RLS).*
//*  This is a prerequisite to using PKI Services SYSPLEX support.    *
//*                                                                    *
//*  The VSAM data sets reallocated by this sample support improved    *
//*  tracking of SCEP requests and certificates issued for those       *
//*  requests that is available in PKI Services V2R3 (HKY77B0) and     *
//*  later releases.  For the sample that reallocates the original     *
//*  VSAM data set format utilized by PKI Services V2R2 (HKY77A0)      *
//*  and earlier releases, please see the IKYRVSAM sample.             *
//*                                                                    *
//**********************************************************************
//*                                                                    *
//*  Caution: This is neither a JCL procedure nor a complete job.      *
//*  Before using this job step, you will have to make the following   *
//*  modifications:                                                    *
//*                                                                    *
//*  1) Change the job card to meet your system requirements.          *
//*                                                                    *
//*  2) Change the STORCLAS statements to provide the name of the      *
//*     storage class defined for use with VSAM RLS.                   *
//*                                                                    *
//*  3) This job assumes you are using the default VSAM data set       *
//*     names (all have high level qualifiers "PKISRVD.VSAM"). If      *
//*     you have changed these data set names, you will need to        *
//*     modify the source data set names in the ALTER                  *
//*     statements of the RENAMEDS step. If you are using              *
//*     multiple CA Domains, IBM recommends using the first eight      *
//*     characters of the CA Domain as one of the data set             *
//*     qualifiers.                                                    *
//*                                                                    *
//*  4) This job creates destination data sets with the same default   *
//*     names as the source data sets. (The source data sets are       *
//*     renamed.) If you wish to use different destination data set    *
//*     names, you will need to modify the data set names in all       *
//*     steps except the RENAMEDS step. If you modify these names,     *
//*     be sure to also modify your configuration file                 *
//*     appropriately(/etc/pkiserv/pkiserv.conf). If you are using     *
//*     multiple CA Domains, IBM recommends using the first eight      *
//*     characters of the CA Domain as one of the data set             *
//*     qualifiers.                                                    *
//*                                                                    *
//*  5) This job renames the source data sets to begin with high       *
//*     level qualifiers "PKISRVD.OLDVSAM". If you wish to change      *
//*     these names, you will need to do so in the RENAMEDS and        *
//*     and REPROCL steps. If you are using multiple CA Domains,       *
//*     IBM recommends using the first eight characters of the CA      *
//*     Domain as one of the data set qualifiers.                      *
//*                                                                    *
//*  6) If you wish to change either the primary or secondary space    *
//*     allocation sizes for either the OST or ICL datasets from the   *
//*     default value, update the CYL or TRK operands on the           *
//*     DEFINE CLUSTER or DEFINE ALTERNATE INDEX commands.             *
//*                                                                    *
//*  **Note, do not change any of the numeric values other than        *
//*        CYL or TRK                                                  *
//*--------------------------------------------------------------------*
//* Rename source clusters, alternate indexes and PATH                 *
//*--------------------------------------------------------------------*
//RENAMEDS  EXEC PGM=IDCAMS
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
    ALTER -
        PKISRVD.VSAM.OST -
        NEWNAME(PKISRVD.OLDVSAM.OST)
    ALTER -
        PKISRVD.VSAM.OST.* -
        NEWNAME(PKISRVD.OLDVSAM.OST.*)
    ALTER -
        PKISRVD.VSAM.ICL -
        NEWNAME(PKISRVD.OLDVSAM.ICL)
    ALTER -
        PKISRVD.VSAM.ICL.* -
        NEWNAME(PKISRVD.OLDVSAM.ICL.*)
    ALTER -
```

```
         PKISRVD.VSAM.OST.AIX.IX -
         NEWNAME(PKISRVD.OLDVSAM.OST.AIX.IX)
     IF LASTCC EQ 8 THEN -
       DO
         SET MAXCC EQ 0
         ALTER -
             PKISRVD.VSAM.AIX.IX -
             NEWNAME(PKISRVD.OLDVSAM.OST.AIX.IX)
       END
     ALTER -
         PKISRVD.VSAM.OST.AIX.DA -
         NEWNAME(PKISRVD.OLDVSAM.OST.AIX.DA)
     IF LASTCC EQ 8 THEN -
       DO
         SET MAXCC EQ 0
         ALTER -
             PKISRVD.VSAM.AIX.DA -
             NEWNAME(PKISRVD.OLDVSAM.OST.AIX.DA)
       END
     ALTER -
         PKISRVD.VSAM.OST.STATAIX.* -
         NEWNAME(PKISRVD.OLDVSAM.OST.STATAIX.*)
     ALTER -
         PKISRVD.VSAM.OST.REQAIX.* -
         NEWNAME(PKISRVD.OLDVSAM.OST.REQAIX.*)
     ALTER -
         PKISRVD.VSAM.OST.SCEPAIX.* -
         NEWNAME(PKISRVD.OLDVSAM.OST.SCEPAIX.*)
     ALTER -
         PKISRVD.VSAM.ICL.STATAIX.* -
         NEWNAME(PKISRVD.OLDVSAM.ICL.STATAIX.*)
     ALTER -
         PKISRVD.VSAM.ICL.REQAIX.* -
         NEWNAME(PKISRVD.OLDVSAM.ICL.REQAIX.*)
     ALTER -
         PKISRVD.VSAM.ICL.SCEPAIX.* -
         NEWNAME(PKISRVD.OLDVSAM.ICL.SCEPAIX.*)
/*
//*-----------------------------------------------------------------*
//* Define destination Clusters                                     *
//*-----------------------------------------------------------------*
//DEFKSDS  EXEC PGM=IDCAMS,COND=(8,LE)
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DEFINE CLUSTER -
        (NAME(PKISRVD.VSAM.OST) -
        STORCLAS(class-name) -
        RECSZ(1024 32756) -
        INDEXED -
        NOREUSE -
        KEYS(4 0) -
        SHR(2) -
        CYL(3,1) -
        LOG(NONE) -
        OWNER(PKISRVD) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.OST.IX))

    DEFINE CLUSTER -
        (NAME(PKISRVD.VSAM.ICL) -
        STORCLAS(class-name) -
        RECSZ(1024 32756) -
        INDEXED -
        NOREUSE -
        KEYS(4 0) -
        SHR(2) -
        LOG(NONE) -
        CYL(3,1) -
        OWNER(PKISRVD) ) -
      DATA -
```

```
        (NAME(PKISRVD.VSAM.ICL.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.ICL.IX))
/*
//*-----------------------------------------------------------------*
//* Repro source cluster to destination cluster                     *
//*-----------------------------------------------------------------*
//REPROCL EXEC PGM=IDCAMS,COND=(8,LE)
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
   REPRO INDATASET(PKISRVD.OLDVSAM.OST) -
      OUTDATASET(PKISRVD.VSAM.OST)
   REPRO INDATASET(PKISRVD.OLDVSAM.ICL) -
      OUTDATASET(PKISRVD.VSAM.ICL)
/*
//*-----------------------------------------------------------------*
//* Define ALTERNATE INDEX AND PATH                                 *
//*-----------------------------------------------------------------*
//DEFALTDX EXEC PGM=IDCAMS,COND=(8,LE)
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.OST.AIX) -
        RELATE(PKISRVD.VSAM.OST)-
        TRK(5,1) -
        KEYS(24 44) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.AIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.OST.AIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.OST.PATH) -
         PATHENTRY(PKISRVD.VSAM.OST.AIX))
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.OST.STATAIX) -
        RELATE(PKISRVD.VSAM.OST)-
        TRK(5,1) -
        KEYS(40 4) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.STATAIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.OST.STATAIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.OST.STATUS) -
         PATHENTRY(PKISRVD.VSAM.OST.STATAIX))
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.ICL.STATAIX) -
        RELATE(PKISRVD.VSAM.ICL)-
        TRK(5,1) -
        KEYS(40 4) ) -
      DATA -
        (NAME(PKISRVD.VSAM.ICL.STATAIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.ICL.STATAIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.ICL.STATUS) -
         PATHENTRY(PKISRVD.VSAM.ICL.STATAIX))
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.OST.REQAIX) -
        RELATE(PKISRVD.VSAM.OST)-
        TRK(5,1) -
        KEYS(32 12) ) -
      DATA -
        (NAME(PKISRVD.VSAM.OST.REQAIX.DA)) -
      INDEX -
        (NAME(PKISRVD.VSAM.OST.REQAIX.IX))
    DEFINE PATH -
        (NAME(PKISRVD.VSAM.OST.REQUESTR) -
         PATHENTRY(PKISRVD.VSAM.OST.REQAIX))
    DEFINE ALTERNATEINDEX -
       (NAME(PKISRVD.VSAM.ICL.REQAIX) -
        RELATE(PKISRVD.VSAM.ICL)-
```

```
         TRK(5,1) -
         KEYS(32 12) ) -
       DATA -
         (NAME(PKISRVD.VSAM.ICL.REQAIX.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.ICL.REQAIX.IX))
     DEFINE PATH -
         (NAME(PKISRVD.VSAM.ICL.REQUESTR) -
          PATHENTRY(PKISRVD.VSAM.ICL.REQAIX))
     DEFINE ALTERNATEINDEX -
        (NAME(PKISRVD.VSAM.OST.SCEPAIX) -
         RELATE(PKISRVD.VSAM.OST)-
         TRK(5,1) -
         KEYS(128 241) ) -
       DATA -
         (NAME(PKISRVD.VSAM.OST.SCEPAIX.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.OST.SCEPAIX.IX))
     DEFINE PATH -
         (NAME(PKISRVD.VSAM.OST.SCEPTID) -
          PATHENTRY(PKISRVD.VSAM.OST.SCEPAIX))
     DEFINE ALTERNATEINDEX -
        (NAME(PKISRVD.VSAM.ICL.SCEPAIX) -
         RELATE(PKISRVD.VSAM.ICL)-
         TRK(5,1) -
         KEYS(128 241) ) -
       DATA -
         (NAME(PKISRVD.VSAM.ICL.SCEPAIX.DA)) -
       INDEX -
         (NAME(PKISRVD.VSAM.ICL.SCEPAIX.IX))
     DEFINE PATH -
         (NAME(PKISRVD.VSAM.ICL.SCEPTID) -
          PATHENTRY(PKISRVD.VSAM.ICL.SCEPAIX))
/*
//*-------------------------------------------------------------------*
//* BUILD ALTERNATE INDEX                                             *
//*-------------------------------------------------------------------*
//BLDINDEX EXEC PGM=IDCAMS,COND=(8,LE)
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.AIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.STATAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.ICL) -
        OUTDATASET(PKISRVD.VSAM.ICL.STATAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.REQAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.ICL) -
        OUTDATASET(PKISRVD.VSAM.ICL.REQAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.OST) -
        OUTDATASET(PKISRVD.VSAM.OST.SCEPAIX)
    BLDINDEX INDATASET(PKISRVD.VSAM.ICL) -
        OUTDATASET(PKISRVD.VSAM.ICL.SCEPAIX)
/*
```

IKYSBIND

IKYSBIND is a sample job to create the Db2 package and plan for the object store and issued certificate list (ICL). IKYSBIND is a member of SYS1.SAMPLIB.

The following listing might not be identical to the code sample shipped with the product. For the most current sample, see SYS1.SAMPLIB member IKYSBIND.

Note: In Version 2 Release 3, there are two DBRMs. One is created with tables following formatting prior to Version 2 Release 3. The other one is following the new format that is introduced in Version 2 Release

3. Ensure that you use the corresponding DBRM to create the corresponding package. IKYPDBRM is the `version 0` DBRM, and IKYPDBR1 is the `version 1` DBRM.

```
//IKYSBIND JOB <job parameters>
//*******************************************************************
//* SAMPLE: IKYSBIND
//*
//* Licensed Materials - Property of IBM
//* 5650-ZOS
//* Copyright IBM Corp.  2011, 2017
//* Status = HKY77B0
//*
//*******************************************************************
//*
//* This sample job may be used to build a DB2 package for
//* PKI Services.
//*
//*******************************************************************
//*
//* This job may need to be modified before it is executed.
//*
//* Before running this job, you need to decide which of the Database
//* Request Modules (DBRMs) provided by PKI Services are to be used
//* for this installation.  PKI Services provides two DBRMs:
//*
//*     IKYPDBRM    Designed to access DB2 database tables created by
//*                 PKI Services Version 2 Release 2 (HKY77A0) and
//*                 earlier.  These databases do not use an unique
//*                 table column to track SCEP Transaction IDs per
//*                 certificate or certificate request.  PKI Services
//*                 installations with existing database tables that
//*                 do not expect to handle many SCEP requests can
//*                 continue to use this DBRM for PKI Servcies
//*                 Version 2 Release 3 and later.
//*
//*                 IKYPDBRM cannot be used to access DB2 databases
//*                 tables created by default for PKI Services
//*                 Version 2 Release 3 (HKY77B0) or later.
//*
//*                 Existing installations of PKI Services need
//*                 IKYPDBRM to allow PKI Services Version 2 Release
//*                 3 to access this information.
//*
//*     IKYPDBR1    Designed to access DB2 database tables created by
//*                 default for PKI Services Version 2 Release 3
//*                 (HKY77B0) or later.  These databases are designed
//*                 for improved tracking of SCEP certificates and
//*                 requests, and implement unique columns to track
//*                 SCEP Transaction IDs.  New installations of PKI
//*                 Services are recommended to use this DBRM.  PKI
//*                 Services installations with existing database
//*                 tables that expect to handle SCEP requests should
//*                 consider converting their existing database tables
//*                 the the Version 2 Release 3 format with the
//*                 "db2conv" utility and making use of this DBRM.
//*
//*                 IKYPDBR1 cannot be used to access DB2 databases
//*                 tables created by the IKYCDB2 sample for PKI
//*                 Services Version 2 Release 2 (HKY77A0) or earlier.
//*
//*                 New installations of PKI Services are recommended
//*                 to use IKYPDBR1.
//*
//*                 Existing installations  of PKI Services that wish
//*                 to exploit the improved SCEP request and
//*                 certificate tracking offered in PKI Services
//*                 Version 2 Release 3 and later releases need
//*                 IKYPDBR1 to access the revised database format.
//*
//* Once you decide what DBRMs are needed for your installation, you
//* will need to package the DBRMs using a variant of the BIND
//* PACKAGE instruction given in this sample.  If you are using both
//* IKYPDBRM and IKYPDBR1, you are recommended to package each DBRM
//* in a separate package.  Existing installations that already have
//* an existing DB2 package/collection for the previous version of
//* IKYPDBRM must package the new IKYPDBRM in the same DB2
//* package/collection.  It is recommended that IKYPDBR1 be packaged
//* in a different DB2 package/collection.
//*
//* Before running this job, you may need to make the following
//* modifications:
//*
//* 1) Change all the occurences of 'MASTERC1' to the package name.
//*    In choosing this name, it is recommended to use the first
//*    seven characters of the associated CA domain name, followed
//*    by a character that will uniquely identify this package name
//*    with the DBRM assigned to this package.  This sample assumes
//*    that 'MASTERCA' is the assocaited CA Domain name, and the
//*    last character '1' is used to associate this package with the
//*    IKYPDBR1 DBRM.  The resulting name used here must match the
//*    value used when creating the DB2 database, tablespaces,
//*    tables, and indices.
//*
//* 2) Change 'DSN9' to the DB2 subsystem name which PKI Services
//*    will be running under.
//*
//* 3) Change 'PKISRVD' to the ID you would like to be the owner of
//*    the package.
//*
//* To issue the following BIND instruction, the user must have at
//* least one of the following DB2 privileges:
//*    - BIND privilege WITH GRANT OPTION on the PACKAGE
//*      specified in the following BIND instruction
//*    - Ownership of the PACKAGE specified in the following
//*      BIND instruction
//*    - SYSADM authority
//*
//*********************************************************************
//EXECTSO  EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//DBRMLIB  DD DSN=SYS1.CBRDBRM,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSIN  DD *
DSN SYSTEM(DSN9)
BIND PACKAGE(MASTERC1)   -
     MEMBER(IKYPDBR1)    -
     QUALIFIER(MASTERC1) -
     VALIDATE(BIND)      -
     ENABLE(RRSAF)       -
     ENCODING(1047)      -
     ACTION(REPLACE)     -
     CURRENTDATA(NO)     -
     DBPROTOCOL(DRDA)    -
     RELEASE(COMMIT)     -
     OWNER(PKISRVD)
END
```

IKYSGRNT

Note: The following listing might not be identical to the code sample shipped with the product. For the most current sample, see SYS1.SAMPLIB member IKYSGRNT.

```
--*********************************************************************
--* SAMPLE: IKYSGRNT
--*
--* Licensed Materials - Property of IBM
--* 5650-ZOS
--* Copyright IBM Corp.  2013, 2017
--* Status = HKY77B0
--*
--*********************************************************************
--
-- This sample may be used to grant execute privilege through SPUFI
-- on the DB2 package for PKI Services to the PKI Services daemon
-- user ID.
--
-- The DB2 package/collection used by this sample assumes that this
-- instance of PKI Services is being configured to use the database
-- format designed for improved tracking of SCEP requests and the
-- certificates issued for those requests that is available in PKI
-- Services Version 2 Release 3 (HKY77B0) and later releases.  This
-- sample assumes that the DB2 package/collection was created using
-- the IKYSBIND sample.
--
--*********************************************************************
--
```

```
-- Before using this sample, you may need to make the following
-- modifications:
--
-- 1) Change 'MASTERC1' to the package name that was used in the
--    IKYSBIND sample JCL file.  In choosing this name, it is
--    recommended to use the first seven characters of the
--    associated CA domain name, followed by a character that will
--    uniquely identify this package name with the DBRM assigned
--    to this package.  This sample assumes that 'MASTERCA' is the
--    assocaited CA Domain name, and the last character '1' is
--    used to associate this package with the IKYPDBR1 package.
--    The resulting name used here must match the value used when
--    creating the DB2 database, tablespaces, tables, and indices
--    in the IKYCDBV1 or IKYCDB2 sample files.
--
-- 2) Change 'PKISRVD' to match the value for the PKI Services daemon
--    user ID name.  The value to use should match the value used for
--    the variable 'daemon' in the IKYSETUP sample file.
--
-- You will also need certain DB2 privileges to use this sample.  To
-- use this sample, you will need at least one of the following DB2
-- privileges:
--      - EXECUTE privilege WITH GRANT OPTION on the
--        package named in the GRANT instruction below
--      - Ownership of the package named in the GRANT
--        instruction below
--      - SYSADM authority
--
--*********************************************************************
GRANT EXECUTE ON PACKAGE MASTERC1.* TO PKISRVD;

COMMIT;
```

IKYVBKUP

IKYVBKUP contains sample JCL to back up the PKI Services VSAM data sets using the DFSMSdss DUMP utility. IKYVBKUP is installed as a member of SYS1.SAMPLIB.

Note: The following listing might not be identical to the code sample shipped with the product. For the most current sample, see SYS1.SAMPLIB member IKYVBKUP.

```
//IKYVBKUP JOB <job card parameters>
//*********************************************************************
//*  SAMP:      IKYVBKUP                                              *
//*                                                                   *
//*    Licensed Materials - Property of IBM                           *
//*    5694-A01                                                       *
//*    Copyright IBM Corp. 2009                                       *
//*    Status = HKY7760                                               *
//*                                                                   *
//*********************************************************************
//*                                                                   *
//*  This sample JCL may be used to backup the VSAM data sets         *
//*  PKI Services utilizes to store certificate requests and          *
//*  issued certificates. To ensure data integrity, the PKI Services  *
//*  address space must be stopped before a backup is attempted.      *
//*                                                                   *
//*********************************************************************
/*                                                                    *
//*  Caution: This is neither a JCL procedure nor a complete job.     *
//*  Before using this job step, you will have to make the following  *
//*  modifications:                                                   *
//*                                                                   *
//*  1) Change the job card to meet your system requirements.         *
//*                                                                   *
//*  2) If you are not using the default data set qualifiers,         *
//*     change all occurrences of "PKISRVD.VSAM" to the qualifiers    *
//*     you are using.                                                *
//*                                                                   *
//*  3) Change all occurrences of vvvvvv to a VOLSER value            *
//*     that contains sufficient free space to contain a complete     *
//*     backup of both of the PKI Services VSAM data set clusters.    *
//*                                                                   *
//*  4) Change the primary and secondary allocation values for the    *
//*     backup dataset to values that will ensure a complete backup   *
//*     of both VSAM data set clusters. Change the xxx value for the  *
//*     primary allocation, and the yyy value for the secondary       *
//*     allocation.                                                    *
//*                                                                    *
//*
//*------------------------------------------------------------------*
//* Delete existing backup dataset
//*------------------------------------------------------------------*
//CATALOG  EXEC PGM=IEHPROGM
//SYSPRINT DD SYSOUT=*
//DD1      DD UNIT=3390,VOLUME=SER=vvvvvv,DISP=OLD,SPACE=(TRK,0)
//SYSIN    DD *
    SCRATCH VOL=3390=vvvvvv,DSNAME=PKISRVD.VSAM.BACKUP
/*
//*------------------------------------------------------------------*
//* Perform a DFSMS/dss Dump of the two PKI Services VSAM clusters
//*------------------------------------------------------------------*
//BACKUP1  EXEC PGM=ADRDSSU,COND=(8,LT)
//BACKUPDS DD DSN=PKISRVD.VSAM.BACKUP,DISP=(NEW,CATLG,DELETE),
//       SPACE=(CYL,(xxx,yyy)),VOL=SER=(vvvvvv)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
    DUMP DATASET(INCLUDE(PKISRVD.VSAM.OST -
                      PKISRVD.VSAM.ICL)) -
         OUTDDNAME(BACKUPDS) -
         CANCELERROR         -
         COMPRESS            -
         OPTIMIZE(4)         -
         SPHERE              -
         WAIT(0,0)           -
         SHR
/*
```

IKYVREST

IKYVREST contains sample JCL to restore the PKI Services VSAM data sets from a backup taken with the DFSMSdss DUMP utility. IKYVREST is installed as a member of SYS1.SAMPLIB.

Note: The following listing might not be identical to the code sample shipped with the product. For the most current sample, see SYS1.SAMPLIB member IKYVREST.

```
//IKYVREST JOB <job card parameters>
//*******************************************************************
//*  SAMP:      IKYVREST                                             *
//*                                                                  *
//*    Licensed Materials - Property of IBM                          *
//*    5694-A01                                                      *
//*    Copyright IBM Corp. 2009                                      *
//*    Status = HKY7760                                              *
//*                                                                  *
//*******************************************************************
//*                                                                  *
//*  This sample JCL may be used to restore the PKI Services VSAM    *
//*  data sets from a backup taken with the DFSMS/dss DUMP utility.  *
//*  The PKI Services address space must be stopped before running   *
//*  this restore job.                                               *
//*                                                                  *
//*******************************************************************
//*                                                                  *
//*  Caution: This is neither a JCL procedure nor a complete job.    *
//*  Before using this job step, you will have to make the following *
//*  modifications:                                                  *
//*                                                                  *
//*  1) Change the job card to meet your system requirements.        *
//*                                                                  *
//*  2) If you are not using the default data set qualifiers,        *
//*     change all occurrences of "PKISRVD.VSAM" to the qualifiers   *
//*     you are using.                                               *
//*                                                                  *
//*  3) If you changed the default data set name for the BACKUPDS    *
//*     DD in the IKYVBKUP jcl, change the dataset name (DSN) value  *
//*     in the BACKUPDS DD statement is this jcl to match the value  *
//*     used in the backup jcl (IKYVBKUP).                           *
//*                                                                  *
//*
//*------------------------------------------------------------------*
//* Perform a DFSMS/dss RESTORE of the two PKI Services VSAM clusters
```

```
//*------------------------------------------------------------------*
//RESTORE1 EXEC PGM=ADRDSSU
//BACKUPDS DD DSN=PKISRVD.VSAM.BACKUP,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
    RESTORE DATASET(INCLUDE(PKISRVD.VSAM.OST,  -
                            PKISRVD.VSAM.ICL)) -
         INDDNAME(BACKUPDS)    -
         CATALOG               -
         CANCELERROR           -
         REPLACEUNCONDITIONAL  -
         SPHERE                -
         WAIT(0,0)
/*
```

PKISERVD sample procedure to start PKI Services daemon

PKISERVD is the sample procedure to start PKI Services daemon. The PKI Services daemon runs as a started task. The procedure for this can be found in `'SYS1.PROCLIB'` member PKISERVD. (PKISERVD is an alias for IKYSPROC.)

PKISERVD contains the **TZ** (time zone) environment variable, which is the environment variable most likely to change. You need to specify any other environment variables that PKI Services needs in an environment variables file, by default `pkiserv.envars`. PKISERVD contains FN (file name) and DIR (directory) parameters to specify the path name of the environment variables file. You can make any needed changes in PKISERVD, such as updating this path name.

Guideline: By default, the path name for the `pkiserv.envars` environment variables file is `/usr/lpp/pkiserv/samples/pkiserv.envars`. If you need to make changes in the environment variables file, you need to copy it from the samples directory to another directory. Specify your environment variables using an environment variables file under the `/etc` directory, for example `/etc/pkiserv/pkiserv.envars`.

The following listing might not be identical to the code sample shipped with the product. For the most current sample, see SYS1.PROCLIB member PKISERVD.

```
//*********************************************************************
//*                                                                   *
//*              Licensed Materials - Property of IBM                 *
//*              5694-A01                                             *
//*              (C) Copyright IBM Corp.  2001                        *
//*              Status=HKY7706                                       *
//*                                                                   *
//*********************************************************************
//*********************************************************************
//*                                                                   *
//* Procedure for starting the PKI Services Daemon                    *
//*                                                                   *
//*********************************************************************
//PKISERVD PROC REGSIZE=256M,                                          X
//             OUTCLASS='A',                                           X
//             TZ='EST5EDT',                                           X
//             FN='pkiserv.envars',                                    X
//             DIR='/usr/lpp/pkiserv/samples',                         X
//             STDO='1>DD:STDOUT',                                     X
//             STDE='2>DD:STDERR'
//*-------------------------------------------------------------------
//GO       EXEC  PGM=IKYPKID,REGION=&REGSIZE,TIME=1440,
//  PARM=('ENVAR("_CEE_ENVFILE=&DIR/&FN","TZ=&TZ") / &STDO &STDE')
//STDOUT   DD  SYSOUT=&OUTCLASS
//STDERR   DD  SYSOUT=&OUTCLASS
//SYSOUT   DD  SYSOUT=&OUTCLASS
//CEEDUMP  DD  SYSOUT=&OUTCLASS
```

Chapter 32. SMF recording

PKI Services produces one SMF record type - type 80. The first 18 bytes of type 80 records represent the standard SMF header without subtypes.

For more information:

1. See *z/OS MVS System Management Facilities (SMF)* for information about how to use SMF.
2. See *z/OS Security Server RACF Macros and Interfaces* and *z/OS Security Server RACF Auditor's Guide* for information about using the RACF SMF data unload utility (IRRADU00) to prepare reports with the RACF report writer.

PKI Services event code

Table 110 on page 683 describes the SMF80EVT (event code) and SMF80EVQ (event code qualifier) fields for the PKI Services event code. It also lists the SMF80DTP and SMF80DA2 values for the relocate type sections.

Table 110. SMF event code and event code qualifier for PKI Services

Event Dec(Hex)	Command	Code qualifier Dec(Hex)	Description	Relocate type sections
79(4F)	CRL publication	0(0)	Successful publication of revocation information	318, 319, 366, 379, 380, 381, 382, 383, 384, 385, 387
85(55)	SUCCRNEW	0(0)	Successful AutoRenew	318, 319, 341, 342, 346, 358, 363, 373, 391, 408

Relocate section variable data

Table 111 on page 683 describes the variable data elements of the extended-length relocate section.

Table 111. SMF data elements of the extended-length relocate section for PKI Services

Data type (SMF80TP2) Dec(Hex)	Data length (SMF80DL2)	Format	Audited by event code	Description (SMF80DA2)
318(13E)	1-255	EBCDIC	66, 67, 69, 72, 74, 79	Certificate or CRL serial number
319(13F)	1-255	EBCDIC	66, 67, 69, 72, 74, 79	Certificate or CRL issuer's distinguished name
366(16E)	4	Binary	74	Certificate revocation reason
379(17B)	1-255	EBCDIC	79	CRL issuing distribution point DN
380(17C)	10	EBCDIC	79	CRL's date of issue
381(17D)	8	EBCDIC	79	CRL's time of issue
382(17E)	10	EBCDIC	79	CRL's expiration date
383(17F)	8	EBCDIC	79	CRL's expiration time
384(180)	10	EBCDIC	79	CRL's date of publish
385(181)	8	EBCDIC	79	CRL's time of publish
387(183)	1-1024	EBCDIC	79	CRL's issuing distribution point URI
408(198)	256	EBCDIC	85	AutoRenew Exit path name

Appendix A. LDAP directory server requirements

PKI Services typically requires access to an LDAP directory server to store issued certificates and certificate revocation lists. The z/OS LDAP server provided by IBM Tivoli Directory Server for z/OS is preferred but not required. You can use a non-z/OS LDAP server if it can support the objectclasses and attributes PKI Services uses. These are listed in the following table:

Table 112. LDAP objectclasses and attributes that PKI Services sets

End-entity or branch node?	Visible RDN attribute	Objectclasses used	Additional attributes set (other than visible RDN attribute)
Creating a branch node	C=	country	—
Creating a branch node	L=	locality	—
Creating a branch node	O=	organization	—
Creating a branch node	OU=	organizationalUnit	—
Creating a branch node	DC=	domain	none
Creating a branch node	Any supported value other than the preceding	organizationalUnit, and extensibleObject	ou (the ou value from CreateOUValue in the **LDAP** section of `pkiserv.conf` file)
Creating a user end-entity	unstructuredName or unstructAddress	account, pkiUser, cEPDevice, and extensibleObject	userCertificate, and uid (hardcoded to `NoUid`)
Creating a user end-entity	serialNumber	account, pkiUser, pKCS10Device, and extensibleObject	userCertificate, and uid (hardcoded to `NoUid`)
Creating a user end-entity	DC	domain pkiUser, , and extensibleObject	userCertificate
Creating a user end-entity	dnQualifier	account, pkiUser, uniquelyQualifiedObject, and extensibleObject	userCertificate, and uid (hardcoded to `NoUid`)
Creating a user end-entity	UID	account, pkiUser, and extensibleObject	userCertificate
Creating a user end-entity	Any supported value other than unstructuredName, unstructAddress, serialNumber, DC, dnQualifier and UID	account, pkiUser, and extensibleObject	userCertificate, and uid (hardcoded to `NoUid`)
Creating a CA end-entity	O=	organization, and pkiCA	cACertificate, certificaterevocationlist, and authorityrevocationlist
Creating a CA end-entity	OU=	organizationalUnit, and pkiCA	cACertificate, certificaterevocationlist, and authorityrevocationlist
Creating a CA end-entity	DC	domain, pkiCA and extensibleObject	cACertificate, certificaterevocationlist, and authorityrevocationlist

Table 112. LDAP objectclasses and attributes that PKI Services sets (continued)

End-entity or branch node?	Visible RDN attribute	Objectclasses used	Additional attributes set (other than visible RDN attribute)
Creating a CA end-entity	dnQualifier	account, uniquelyQualifiedObject, pkiCA, and extensibleObject	cACertificate, certificaterevocationlist, authorityrevocationlist, and uid (hardcoded to `NoUid`)
Creating a CA end-entity	UID	account, pkiCA, and extensibleObject	cACertificate, certificaterevocationlist, and authorityrevocationlist
Creating a CA end-entity	Any supported value other than O, OU, DC, dnQualifier and UID	account, pkiCA, and extensibleObject	cACertificate, certificaterevocationlist, authorityrevocationlist and uid (hardcoded to `NoUid`)
User end-entity that already exists	unstructuredName or unstructAddress	pkiUser, cEPDevice	userCertificate
User end-entity that already exists	serialNumber	pkiUser, pKCS10Device	userCertificate
User end-entity that already exists	Any supported value other than unstructuredName, unstructAddress, and serialNumber	pkiUser	userCertificate
CA end-entity that already exists	Any supported value	pkiCA	cACertificate, certificaterevocationlist, and authorityrevocationlist
Creating a distribution point CRL end-entity	CN=	commonName and cRLDistributionPoint	certificateRevocationList
Distribution point CRL end-entity that already exists	Any supported value	cRLDistributionPoint	certificateRevocationList

The `R_PKIServ` SAF callable service supports specifying the subject's DN through named fields in the CertPlist. The CGIs invoke the `R_PKIServ` SAF callable service. For more information, see *z/OS Security Server RACF Callable Services*. PKI Services supports the subject's DN fields, plus some additional ones: postal code, street, and mail. They are mapped to LDAP attributes as Table 113 on page 686 indicates.

Table 113. Relationship of named fields to LDAP attributes and object identifiers

Named field	Visible RDN attribute	OID
CommonName	CN	2.5.4.3
Title	TITLE	2.5.4.12
OrgUnit	OU	2.5.4.11
Org	O	2.5.4.10
Locality	L	2.5.4.7
StateProv	ST	2.5.4.8
Country	C	2.5.4.6

Table 113. Relationship of named fields to LDAP attributes and object identifiers (continued)

Named field	Visible RDN attribute	OID
PostalCode	POSTALCODE	2.5.4.17
Street	STREET	2.5.4.9
Email[1]	MAIL	0.9.2342.19200300.100.1.3
Mail	MAIL[2]	0.9.2342.19200300.100.1.3
EmailAddr	EMAIL	1.2.840.113549.1.9.1
UnstructName	UNSTRUCTUREDNAME	1.2.840.113549.1.9.2
UnstructAddr	UNSTRUCTUREDADDRESS	1.2.840.113549.1.9.8
SerialNumber	SERIALNUMBER	2.5.4.5
DNQualifier	DNQUALIFIER	2.5.4.46
DomainName	DC	0.9.2342.19200300.100.1.25
Uid	UID	0.9.2342.19200300.100.1.1
BusinessCat	BUSINESSCATEGORY	2.5.4.15
JurLocality	JURISDICTIONLOCALITY	1.3.6.1.4.1.311.60.2.1.1
JurStateProv	JURISDICTIONSTATEPROV	1.3.6.1.4.1.311.60.2.1.2
JurCountry	JURISTDICTIONCOUNTRY	1.3.6.1.4.1.311.60.2.1.3

[1] The use of the field name Email is deprecated; use Mail instead. [2] When a certificate is created and posted to LDAP, the NotifyEmail value, if specified, is posted as the MAIL attribute. (This replaces any MAIL attribute for the directory entry and for certificate renewals replaces the original NotifyEmail value).

Appendix B. Using a gskkyman key database for your certificate store

This topic lists the steps the RACF programmer performs to use a gskkyman key database.

Steps for using a gskkyman key database for your certificate store

Perform the following steps to use a gskkyman key database for your server's certificate store:

Note: If the IBM HTTP Server is installed and configured for SSL using gskkyman, you need to perform only steps "9" on page 689, "10" on page 690, "11" on page 690, and "15" on page 690.

1. From the UNIX shell, **cd** to `/etc` and enter `/usr/lpp/gskssl/bin/gskkyman`.

   ____

2. Choose option **1** to create a key database. Type in a name or let it default to `key.kdb` and enter a password you want to use. When asked "Work with the database now?", enter **1** for yes.

   ____

3. Choose option **3** to create new key pair and certificate request. Answer the prompts for file name, label, key size (1024 is suggested), and subject name fields.

   Note: Common Name should be your server's symbolic IP address (for example, www.*YourCompany*.com).

   ____

4. Exit gskkyman when you are done.

   ____

5. From TSO, use the OGET command to put the certificate request in an MVS data set.

   Example:

   ```
   OGET '/etc/certreq.arm' certreq.arm
   ```

   ____

6. Use the RACDCERT command to read the request and generate the server certificate.

   Example:

   ```
   RACDCERT GENCERT(certreq.arm) ID(WEBSRV) SIGNWITH(CERTAUTH
   LABEL('Local PKI CA')) WITHLABEL('SSL Cert')
   ```

   ____

7. Export both the new server certificate and the CA certificate to MVS data sets, and OPUT these to file system files.

   Example:

   ```
   RACDCERT EXPORT(LABEL('SSL Cert')) ID(WEBSRV) DSN(cert.arm)
   FORMAT(CERTB64)
   OPUT cacert.der '/var/pkiserv/cacert.der' BINARY
   ```

   ____

8. You can optionally delete both certificate TSO data sets (but not the file system files).

   ____

9. In the UNIX shell, **cd** to `/etc` and invoke `/usr/lpp/gskssl/bin/gskkyman`.

____

10. Choose option **2** to open the key database (created earlier). Reply to the name and password prompts.

    ____

11. Choose option **6** to store a CA certificate and specify the `'/var/pkiserv/cacert.der'` file.

    ____

12. When asked to "Exit gskkyman?", enter **0** for No.

    ____

13. Choose option **4** to receive a certificate issued for your request and specify the `'/etc/cert.arm'` file. Again enter `0` when asked to "Exit gskkyman?".

    ____

14. Choose option **11** to store encrypted database password.

    ____

15. Exit gskkyman.

    ____

16. You can optionally remove the `/etc/cert.arm` file.

    ____

Appendix C. Using the PKI Services web application with Internet Explorer on Windows systems

To use the PKI Services end-user web application through the Internet Explorer browser on a Microsoft Windows system, a user must first set up the Windows system and Internet Explorer to work with PKI Services. This topic describes the tasks that a user needs to perform.

Note: Users do not need to perform these tasks for browsers other than Internet Explorer.

To avoid loss of function, PKI Services provides ActiveX programs that can be used to provide the function that PKI Services requires to renew a certificate. The PKI Services administrator must decide whether to provide signed or unsigned versions of the ActiveX programs, and in which directory the ActiveX programs are stored on the PKI Services server. These administrator tasks are described in "Administrator tasks for setting up a Windows system and Internet Explorer to work with the PKI Services web application" on page 695.

The script that is used in the template and JavaServer page (JSP) to renew and install a certificate from Internet Explorer calls the PKI Services ActiveX program for the required functions.

User tasks for setting up a Windows system and Internet Explorer to work with the PKI Services web application

Table 114. Tasks to perform to set up a Windows system and the Internet Explorer browser to work with PKI Services

Task	**Associated instructions**
Install the PKI Services ActiveX program for your version of Microsoft Windows.	"Installing the PKI Services ActiveX program" on page 691
Configure Internet Explorer to trust PKI Services. (All Windows systems.)	"Configuring Internet Explorer to trust PKI Services on a Windows system" on page 694
Install the z/OS PKI Services certificate authority (CA) certificate on the Microsoft Windows system.	"Installing the PKI Services CA certificate on a Microsoft Windows system" on page 694

Installing the PKI Services ActiveX program

There are two ways to install the PKI Services ActiveX program for your version of Windows:

- From the PKI Services home page (see "Steps for installing the PKI Services ActiveX program from the PKI Services home page" on page 691).
- From the certificate renewal page when you renew a certificate (see "Steps for installing the PKI Services ActiveX program when you renew a certificate" on page 693). If you do not have the PKI Services ActiveX program for your version of Microsoft Windows installed, PKI Services prompts you to install it.

Steps for installing the PKI Services ActiveX program from the PKI Services home page

Perform the following steps to install the PKI Services ActiveX program for your version of Windows from the PKI Services home page.

Before you begin

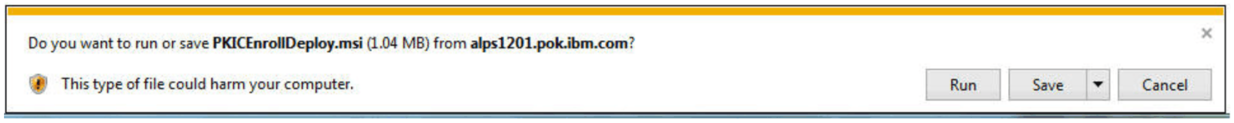
You must run as the system administrator on the Microsoft Windows system on which you are installing the PKI Services ActiveX program, and you must be using the Internet Explorer browser.

Procedure

1. On the PKI Services home page ("#unique_66/unique_66_Connect_42_mainpage" on page 376), click **Install the PKI ActiveX Control to renew certificates**

   ____

2. A file download window opens.

   Do you want to run or save **PKICEnrollDeploy.msi** (1.04 MB) from **alps1201.pok.ibm.com**?

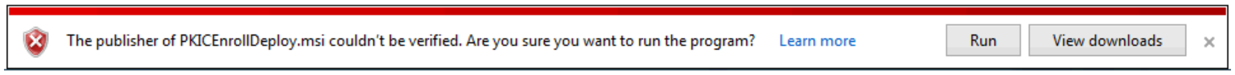
   This type of file could harm your computer.

   Figure 88. File download window

   Click **Run**.

   ____

3. A security warning window opens. If the ActiveX program is signed, the name of the signer appears in the **Publisher** field. Figure 89 on page 692 shows the security warning window for an unsigned ActiveX program. (The PKI Services administrator decides whether or not to sign the ActiveX program.)

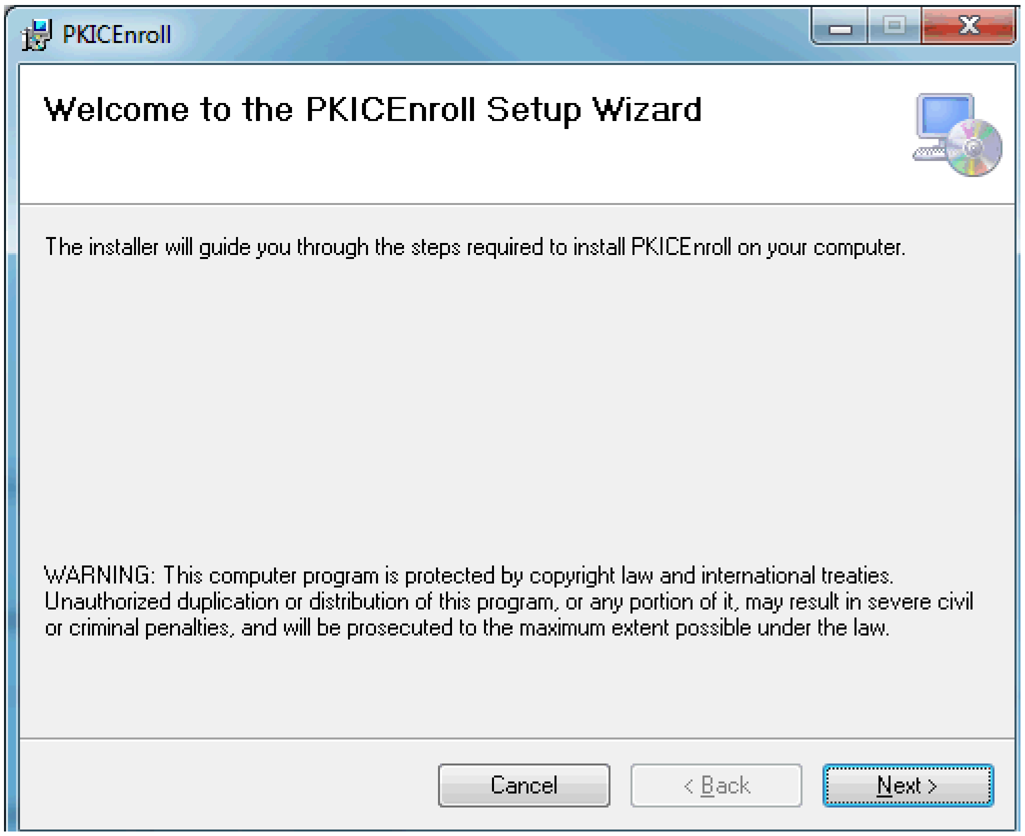
   The publisher of PKICEnrollDeploy.msi couldn't be verified. Are you sure you want to run the program?

   Figure 89. Security warning window

   Click **Run**.

   ____

4. The setup wizard for the PKI ActiveX control opens.

   Welcome to the PKICEnroll Setup Wizard

   The installer will guide you through the steps required to install PKICEnroll on your computer.

   WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

   Figure 90. ActiveX control setup wizard

Click **Next**, and continue clicking **Next** on the windows that follow until you reach **Finish**. Click **Finish** to complete the installation.

____

Results

When you are done, you have installed the PKI Services ActiveX program on a Microsoft Windows system. Continue to the next task, "Configuring Internet Explorer to trust PKI Services on a Windows system" on page 694.

Steps for installing the PKI Services ActiveX program when you renew a certificate

Perform the following steps to install the PKI Services ActiveX program for your version of Windows from the certificate renewal page.

Before you begin

You must run as the system administrator on the Microsoft Windows system on which you are installing the PKI Services ActiveX program.

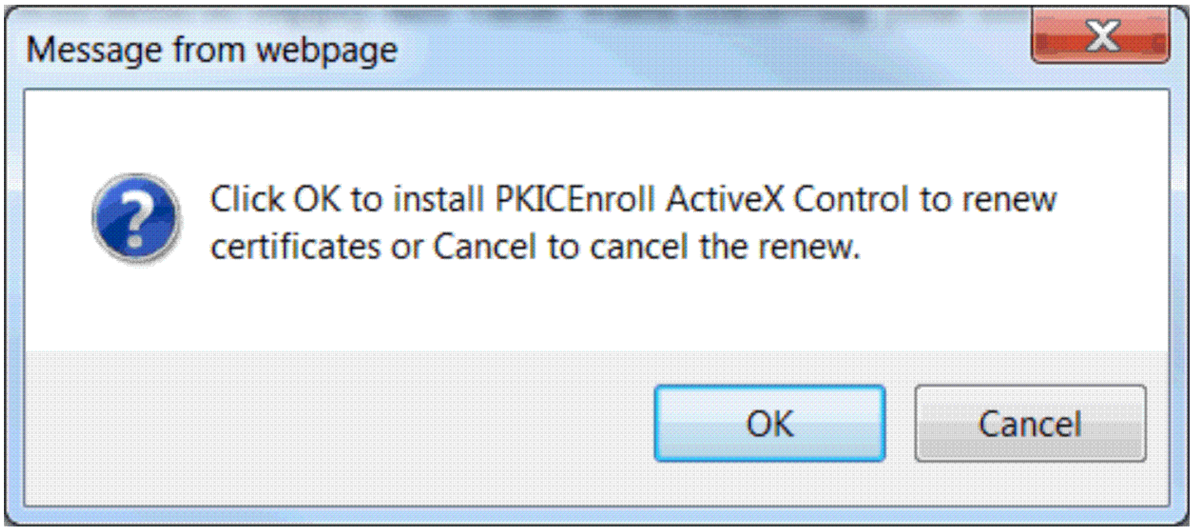
Procedure

1. On the certificate renewal page (Figure 55 on page 397), click **Renew**.

   ____

2. If PKI Services determines that the PKI Services ActiveX program is not installed:

   a) A message window opens.

   Message from webpage

   Click OK to install PKICEnroll ActiveX Control to renew certificates or Cancel to cancel the renew.

   Figure 91. Message window

   Click **OK** to install the ActiveX program.

   b) Continue with the steps for installing the ActiveX control from the PKI Services home page, beginning with step "2" on page 692.

3. Refresh the certificate renewal web page to load the ActiveX control on the browser.

   ____

Results

When you are done, you have installed the PKI Services ActiveX program on a Microsoft Windows system from the certificate renewal web page, and loaded it on the Internet Explorer browser. PKI Services continues with the renewal of your certificate, using the ActiveX program that you just installed.

Configuring Internet Explorer to trust PKI Services on a Windows system

To request or administer certificates using the PKI Services end-user web application through Internet Explorer, you must add the PKI Services system to the list of trusted sites recognized by Internet Explorer.

Steps for configuring Internet Explorer to trust PKI Services
Perform the following steps to configure Internet Explorer to trust PKI Services.

Procedure

1. From the Microsoft Windows system, start the Internet Explorer browser.

   ____

2. Click `Tools->Internet Options` to see the "Internet Options" panel, and select the "Security" tab.

   ____

3. Select the "Trusted sites" icon in the window labeled "Select a zone to view or change security settings". The "Trusted sites" icon is usually represented as a green check mark.

   ____

4. If necessary, click "Custom Level..." to change the settings for the zone. Set "Security level" to `Medium` and ensure that "Enable protected mode (requires restarting Internet Explorer)" is not selected.

   ____

5. Click **Sites** to see the "Trusted Sites" panel.
6. In the area labeled "Add this website to the zone:" enter the URL for the PKI Services system. Use `https` as the protocol, not `http`. For example, if the URL for the PKI Services system is `alps4049.pok.ibm.com`, enter `https://alps4049.pok.ibm.com`. Click **Add** to add this site to the list of trusted sites.

   ____

7. Leave the box labeled "Require server verification (https:) for all sites in this zone" checked. Click **Close** to close the "Trusted Sites" panel and return to the "Security" tab of the "Internet Options" panel.

   ____

8. Click **Apply** on the "Internet Options" panel to confirm the configuration changes. Then click **OK** to close the" Internet Options" panel.

   ____

9. Shut down the Internet Explorer browser to allow the modifications to take effect.

   ____

Results

When you are done, you have configured Internet Explorer to trust PKI Services. Continue to the next task, “Installing the PKI Services CA certificate on a Microsoft Windows system” on page 694.

Installing the PKI Services CA certificate on a Microsoft Windows system

Any system using Microsoft Windows to request or administer certificates using the PKI Services end-user web application through Internet Explorer must install the z/OS PKI Services certificate authority (CA) certificate to enable SSL-protected sessions. Although the Internet Explorer browser can import the z/OS PKI Services CA certificate, the browser does not correctly install the certificate in the proper location by default.

Steps for installing the PKI Services CA certificate on a Microsoft Windows system

Perform the following steps to install the z/OS PKI Services CA certificate in the correct location using Internet Explorer.

Before you begin

You need to know what version of Windows is running on your system.

Procedure

1. Obtain the file that contains the CA certificate with file extension ".cer". For more information, see "Setting up IBM HTTP Server - Powered by Apache" on page 101.

   ____

2. Double click on the file. Internet Explorer presents a "Certificate" pop-up panel indicating that this CA Root certificate is not trusted, and to enable the trust, the certificate must be installed in the Trusted Root Certification Authorities store. Click **Install Certificate**.

   ____

3. Internet Explorer presents a " Certificate Import Wizard " welcome panel. Click **Next >**.

   ____

4. Internet Explorer presents a "Select Certificate Store" pop-up panel, allowing you to select the proper certificate store. Select the "Trusted Root Certification Authorities" item in the scrolled selection window, and click **OK**. Then click **Next >**.

   ____

5. Click **Finish**.

   ____

6. Internet Explorer presents a "Security Warning" pop-up panel, indicating that a certificate is about to be installed, and asking you to verify that this is the intended action to take. Click **Yes**. Internet Explorer presents a confirmation pop-up, indicating that the certificate was successfully imported.

   ____

Results

When you are done, you have successfully installed the PKI Services CA certificate, and your Windows system and Internet Explorer browser are ready to use the PKI Services web application.

Administrator tasks for setting up a Windows system and Internet Explorer to work with the PKI Services web application

PKI Services provides an ActiveX program to provide function that PKI Services requires to install a renewed certificate. The program is:

- PKICEnroll

The ActiveX program requires the Microsoft C Runtime Library and the Microsoft Active Template Library. PKI Services provides a Microsoft installer program for the ActiveX program that is packaged with the ActiveX program and the required Microsoft libraries. You can use the program in one of two ways:

- If you choose not to sign the ActiveX program, you can use the installer program as it is shipped.
- For greater security, you can sign the ActiveX program, but if you do this you must repackage the installer program. PKI Services provides the related registry files, type library files, and the license file that you need to sign and repackage the program. After signing and repackaging the program, you must put the .exe and .msi files in a directory on the PKI Services server that is accessible to PKI Services users so that they can install the ActiveX program.

Guideline: For maximum security, sign the ActiveX program with a certificate issued by your PKI Services CA. The ability of ActiveX program to modify your system makes them a security risk. Signing your ActiveX program helps to ensure that users are running unchanged versions free of viruses.

Signing the PKI Services ActiveX programs

To sign a PKI Services ActiveX program, you must sign the unsigned version that PKI Services provides, rebuild the install program for it, and sign the rebuilt install program. To make the signed programs available to PKI Services users, you must put the signed files in directories that are accessible to PKI Services users, and update the HTTP Server configuration files to specify those directories.

Steps for signing the PKI Services ActiveX programs

Perform the following steps to sign the ActiveX program that PKI Services provides and make it available to PKI Services users.

Before you begin

- You need to have a tool such as Microsoft Visual Studio that builds an installer program.
- You need a code signing certificate with an Extended Key Usage of Code Signing and its private key. If you do not have one, you can request one from PKI Services using the 2-year Authenticode template. Follow the instructions for requesting a server certificate in "Steps for requesting a new certificate" on page 384. Then export the certificate and its private key to a PKCS #12 file and download it to the Windows platform. In step "2.a" on page 696, Microsoft Sign Tool uses the certificate to sign `PKICEnroll.dll`.

Procedure

1. Create directories on the PKI Services server for the .exe and .msi program that you build in step "2.a" on page 696.

   ____
2. Sign the ActiveX program. Perform the following step for `PKICEnroll.dll`.
   a) Use the Microsoft Sign Tool (msdn.microsoft.com/en-us/library/aa387764(VS.85).aspx) (`signtool.exe`) to sign the ActiveX program (`PKICEnroll.dll`) with the code signing certificate.
   b) Build the installer programs. You need to use a tool such as Microsoft Visual Studio.

      Use PKICEnrollDeploy as the project name. The outputs for the installer program are:

      - `setup.exe`
      - `PKICEnrollDeploy.msi`

      For detailed instructions, see "Steps for building the installer programs using Microsoft Visual Studio" on page 697.
   c) Use Microsoft Sign Tool and the code signing certificate that you used in step "2.a" on page 696 to sign the installer programs.

   ____
3. Upload the signed installer programs to the directories you created on the PKI Services server in step "1" on page 696.

   Note: Be sure that you upload the programs in binary so that the files are not altered during the transfer.

   ____
4. Update the HTTP configuration files.
   a) Update `httpd.conf`. Change the following statements to specify the directories you created in step "1" on page 696.

      ```
      Pass /PKIServ/PKICEnroll/*   /usr/lpp/pkiserv/ActiveX/PKICEnroll/*
      ```

b) Update `httpd2.conf`. Change the following statements to specify the directories you created in step "1" on page 696.

```
Pass /PKIServ/PKICEnroll/*   /usr/lpp/pkiserv/ActiveX/PKICEnroll/*
```

____

Results

When you are done, you have signed the ActiveX program provided by PKI Services, and made it available to PKI Services users.

Steps for building the installer programs using Microsoft Visual Studio

Before you begin

You need to have a tool such as Microsoft Visual Studio that builds an installer program.

About this task

These instructions describe how to use Microsoft Visual Studio Software Application Version 2010 to create a setup and deployment Visual Studio project. The project creates a directory to save the project setup files along with the output files.

Perform the following steps to build the installer programs for a PKI Services ActiveX program.

Procedure

1. Download the following files from the directory `/usr/lpp/pkiserv/ActiveX/signsrc` on the z/OS system to a working directory on your workstation:
   - PKICEnroll.dll
   - PKICEnroll.tlb
   - PKICEnroll.reg
   - PKIActiveX.lic

   Note: Ensure that the files are transferred in binary format so that they are not modified in transit.

   ____
2. Open Microsoft Visual Studio Software Application Version 2010 from the Windows Start menu, and create a new setup and deployment project.
   a) Click **File > New Project**.
   b) In the New Project panel, under **Other Project Types** select **Setup and Deployment**.
   c) In the "Visual Studio installed templates" pane (on the right) click **Setup Project**.
   d) In the **Name** field, enter the name of the project, for example PKICEnrollDeploy.
   e) In the **Location** field, enter the directory where you want the project created, or click **Browse** to select a directory.
   f) Select **Create directory for solution**.
   g) Click **OK**.
   A new project directory is created in a separate file directory in the path that is given in the **Location** field.

   ____
3. Add the required ActiveX Dynamic link library, the ActiveX Type library file, and the ActiveX license file (PKIActiveX.lic) to the current project to create the installer program.
   a) Click **View > Solution Explorer**.
   b) Right click the name of the project that you just created, (for example, PKICEnrollDeploy).

c) Click **Add > File**.

d) On the AddFiles window, navigate to the directory where the files were stored on the workstation in step "1" on page 697.

   For PKICEnroll select these files:

   - PKICEnroll.dll
   - PKICEnroll.tlb
   - PKIActiveX.lic

   Click **Open**. The files you selected are listed under the project on Solution Explorer. The DLL file has a dependency on Microsoft .NET Framework. Microsoft Visual Studio automatically lists the dependency under the project.

____

4. Add the Microsoft C Runtime Library and Microsoft Active Template Library merge modules to be packaged in the installer program.

   a) On the Solution Explorer pane, right click the project name (for example, PKICEnrollDeploy).

   b) Click **Add > Merge Module**. A window opens listing all the merge modules that were installed when Microsoft Visual Studio was installed.

   c) Click `Microsoft_VC100_CRT_x86.msm` and `Microsoft_VC100_ATL_x86.msm` and click **Open** to add these files to the project.

____

5. Modify the project properties.

   a) From the Solution Explorer pane, click the project name (for example, PKICEnrollDeploy).

   b) On the toolbar click **View > Other Windows > Property Window**. A list of properties with default values is displayed.

   c) For the Author property, enter `IBM`.

   d) For the InstallAllUsers property, enter `True`.

   e) For the Manufacturer property, enter `IBM`.

   f) For the ProductName property, enter `PKICEnroll`.

   g) For the RemovePreviousVersions property, enter `True`.

____

6. Determine the default location where the ActiveX program is going to be installed.

   a) Click **View > Solution Explorer**.

   b) Right click the project (for example, PKICEnrollDeploy).

   c) Click **View > File System**. A file system pane opens on the right side.

   d) In the File System pane, right click **Application Folder**.

   e) Click **Properties Window**.

   f) Note the value listed in the **Default Location** field. This is the location where the ActiveX program is going to be installed. The default value is *[ProgramFilesFolder][Manufacturer]\[ProductName]*. The value of *ProgramFilesFolder* has been set by Microsoft Visual Studio to the Program Files folder for the operating system: `C:\Program Files` for a 32-bit Windows system and `C:\Program Files(x86)` for a 64-bit operating system. You set the value of *Manufacturer* to IBM and the value of *ProductName* to PKICEnroll when you set the project properties in step "5" on page 698. Do not modify any of these values. The ActiveX DLL looks for the license file (PKIActive.lic) in this directory. If it cannot find it there, the ActiveX program is not instantiated on the browser and certificate renewal processing might not work properly.

____

7. The User Interface command provides the interface for the user during installation. It allows the user installing the ActiveX program to select a directory for installation. Disable the folder selection step, so that the location listed in step "6.f" on page 698 is used.
   a) Click **View > Solution Explorer**.
   b) Right click the project (for example, PKICEnrollDeploy).
   c) Click **View > User Interface**. A User Interface pane opens on the right side.
   d) Under **Install**, click **Installation Folder** and click **Delete**.
   e) Under **Install**, under **End** right click **Finished** and click **Properties**. Modify the UpdateText property to include instructions to be displayed to the user after the installation of the ActiveX program: `Please refresh the PKI Certificate Renewal web page to use the newly installed PKI ActiveX Control.`
   f) Under **Administrative Install**, click **Installation Folder** and click **Delete**.
   g) Under **Administrative Install**, under **End** right click **Finished** and click **Properties**.

   ____
8. The setup and deployment project (for example, PKICEnrollDeploy) can create registry entries for the ActiveX program. Once the ActiveX program is installed on the target machine the ActiveX program is registered and the browsers accessing this ActiveX program instantiates looking at the windows registry. The registry entries are created using the Registry setup interface.
   a) Click **View > Solution Explorer**.
   b) Right click the project (for example, PKICEnrollDeploy).
   c) Click **View > Registry**. A Registry pane opens on the right side.
   d) Right click **Registry on Target machine** .
   e) Click **Import**. The Import Registry File window opens.
   f) Click PKICEnroll.reg and click **Open**.

____
9. Set the ActiveX dynamic link library (PKICEnroll.dll) to be registered during installation.
   a) Click **View > Solution Explorer**.
   b) Right click PKICEnroll.dll.
   c) Click **Properties**.
   d) Set the **Register** field to `vsdraCOM`.

____
10. Set the ActiveX type library (PKICEnroll.tlb) to be registered during install.
    a) Click **View > Solution Explorer**.
    b) Right click PKICEnroll.tlb.
    c) Click **Properties**.
    d) Set the **Register** field to `vsdrfCOM`.

____
11. Build the setup and deployment project.
    a) Click **View > Solution Explorer**.
    b) Right-click the project name from the Solution Explorer (for example, PKICEnrollDeploy).
    c) Select Properties. The **Property Pages** window opens.
    d) Select Release in the Configuration list.
    e) On the right pane of the window that opens, click **Prerequisites**.
    f) Select the **Create setup program to install prerequisite components** check box.

g) In the list of prerequisites, select the **.NET Framework 4.0 Client Profile (x86 and x64)** check box if it is not already selected.

h) Select the **Visual C++ 2010 Runtime Libraries (x86)** check box.

i) Click **OK**. The **Prerequisites** window closes.

j) Click **OK**. The **Property Pages** window closes.

k) Right-click the project (for example, PKICEnrollDeploy) and select **Build**.

If the build is successful, the application creates an output directory under the root project directory that is under the main solution directory. If PKICEnrollDeploy is the project name:

- C:\PKICEnrollDeploy is the solution name.
- C:\PKICEnrollDeploy\PKICEnrollDeploy is the root project directory.
- C:\PKICEnrollDeploy\PKICEnrollDeploy\Release is the output directory.
- There are two output files: PKICEnrollDeploy.msi and setup.exe.

Results

When you are done, you have built the installer program for a PKI Services ActiveX program. Continue with step “2.c” on page 696 to sign the installer program.

Appendix D. Accessibility

Accessible publications for this product are offered through IBM Knowledge Center (www.ibm.com/support/knowledgecenter/SSLTBW/welcome).

If you experience difficulty with the accessibility of any z/OS information, send a detailed message to the Contact the z/OS team web page (www.ibm.com/systems/campaignmail/z/zos/contact_z) or use the following mailing address.

IBM Corporation
Attention: MHVRCFS Reader Comments
Department H6MA, Building 707
2455 South Road
Poughkeepsie, NY 12601-5400
United States

Accessibility features

Accessibility features help users who have physical disabilities such as restricted mobility or limited vision use software products successfully. The accessibility features in z/OS can help users do the following tasks:

- Run assistive technology such as screen readers and screen magnifier software.
- Operate specific or equivalent features by using the keyboard.
- Customize display attributes such as color, contrast, and font size.

Consult assistive technologies

Assistive technology products such as screen readers function with the user interfaces found in z/OS. Consult the product information for the specific assistive technology product that is used to access z/OS interfaces.

Keyboard navigation of the user interface

You can access z/OS user interfaces with TSO/E or ISPF. The following information describes how to use TSO/E and ISPF, including the use of keyboard shortcuts and function keys (PF keys). Each guide includes the default settings for the PF keys.

- *z/OS TSO/E Primer*
- *z/OS TSO/E User's Guide*
- *z/OS ISPF User's Guide Vol I*

Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users who access IBM Knowledge Center with a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line because they are considered a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or `3.1` or `3.1.1`. To hear these numbers correctly, make sure that the screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number `3.1`) are mutually exclusive alternatives. If you hear the lines `3.1 USERID` and `3.1 SYSTEMID`, your syntax can include either `USERID` or `SYSTEMID`, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The * symbol is placed next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element `*FILE` with dotted decimal number 3 is given the format `3 \* FILE`. Format `3* FILE` indicates that syntax element `FILE` repeats. Format `3* \* FILE` indicates that syntax element `* FILE` repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol to provide information about the syntax elements. For example, the lines `5.1*, 5.1 LASTRUN, and 5.1 DELETE` mean that if you use more than one of the `LASTRUN` and `DELETE` syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, it indicates a reference that is defined elsewhere. The string that follows the % symbol is the name of a syntax fragment rather than a literal. For example, the line `2.1 %OP1` means that you must refer to separate syntax fragment OP1.

The following symbols are used next to the dotted decimal numbers.

? indicates an optional syntax element

The question mark (?) symbol indicates an optional syntax element. A dotted decimal number followed by the question mark symbol (?) indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example `5? NOTIFY`). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines `5 ?, 5 NOTIFY`, and `5 UPDATE`, you know that the syntax elements `NOTIFY` and `UPDATE` are optional. That is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.

! indicates a default syntax element

The exclamation mark (!) symbol indicates a default syntax element. A dotted decimal number followed by the `!` symbol and a syntax element indicate that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the dotted decimal number can specify the ! symbol. For example, if you hear the lines `2? FILE, 2.1! (KEEP), and 2.1 (DELETE)`, you know that `(KEEP)` is the default option for the `FILE` keyword. In the example, if you include the `FILE` keyword, but do not specify an option, the default option KEEP is applied. A default option also applies to the next higher dotted decimal number. In this example, if the `FILE` keyword is omitted, the default `FILE(KEEP)` is used. However, if you hear the lines `2? FILE, 2.1, 2.1.1! (KEEP)`, and `2.1.1 (DELETE)`, the default option KEEP applies only to the next higher dotted decimal number, `2.1` (which does not have an associated keyword), and does not apply to `2? FILE`. Nothing is used if the keyword `FILE` is omitted.

*** indicates an optional syntax element that is repeatable**

The asterisk or glyph (*) symbol indicates a syntax element that can be repeated zero or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line `5.1* data area`, you know that you can include one data area, more than one data area, or no data area.

If you hear the lines `3* , 3 HOST, 3 STATE`, you know that you can include `HOST, STATE`, both together, or nothing.

Notes:

1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you can write `HOST STATE`, but you cannot write `HOST HOST`.
3. The * symbol is equivalent to a loopback line in a railroad syntax diagram.

+ indicates a syntax element that must be included
The plus (+) symbol indicates a syntax element that must be included at least once. A dotted decimal number followed by the + symbol indicates that the syntax element must be included one or more times. That is, it must be included at least once and can be repeated. For example, if you hear the line `6.1+ data area`, you must include at least one data area. If you hear the lines `2+, 2 HOST, and 2 STATE`, you know that you must include `HOST, STATE,` or both. Similar to the * symbol, the + symbol can repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loopback line in a railroad syntax diagram.

Notices

Terms and conditions for product documentation

Applicability

Personal use

Commercial use

reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user’s name, email address, phone number, or other personally identifiable information for purposes of enhanced user usability and single sign-on configuration. These cookies can be disabled, but disabling them will also eliminate the functionality they enable.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM’s Privacy Policy at ibm.com/privacy and IBM’s Online Privacy Statement at ibm.com/privacy/details in the section entitled “Cookies, Web Beacons and Other Technologies,” and the “IBM Software Products and Software-as-a-Service Privacy Statement” at ibm.com/software/info/product-privacy.

Policy for unsupported hardware

Various z/OS elements, such as DFSMSdfp, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Minimum supported hardware

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: IBM Lifecycle Support for z/OS (www.ibm.com/software/support/systemsz/lifecycle)
- For information about currently-supported IBM hardware, contact your IBM representative.

Programming interface information

This document primarily documents information that is NOT intended to be used as Programming Interfaces of PKI Services.

This document also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of PKI Services. This information is identified where it occurs, either by an introductory statement to a topic or section or by the following marking:

Programming Interface Information

End Programming Interface Information

Trademarks

Index

Product Number: 5650-ZOS

SA23-2286-40