Hynetd v0.2.4 Manual

(**Hy**brid **Ne**twork **T**opology **D**iscovery)

Alessio Botta, Walter de Donato and Antonio Pescapè
University of Naples Federico II

January 24, 2007

Index

1 Required libraries 3

2 License 3

3 Installation 3

4 How to use it 3

4.1 Options description 4

5 Examples 6

5.1 Example 1 6

5.2 Example 2 6

5.3 Example 3 6

1 Required libraries

Hynetd requires these libraries to compile and work correctly:

- **net-snmp >= 5.2**
- **libpthread**

2 License

This program is open-source and completely free for Universities, Schools, Research Institutes, and Individuals; it is also free for Companies that use HyNeTD for their internal research; you can redistribute it and/or modify it under the terms of the HYNETD NON-COMMERCIAL LICENSE as reported in the LICENSE file of this ditribution; either version 1 of the License, or (at your option) any later version.

For commercial information please contact us to our email.

3 Installation

The easiest way to compile hynetd is launching the following command:

```
bash$ make hynetd
```

read INSTALL for further details...

4 How to use it

First of all, root privileges are needed to use this tool, because it uses standard raw sockets to manage ICMP packets.

Hynetd also needs at least 3 parameters to be defined on command line:

```
-t <max_number_of_threads>      number of threads to be created during scan
<base_address>                  base IP address of the first range
<address_space_size>            amplitude of the first range
```

The correct syntax to be used can be displayed using '-h' option.

```
HyNeTD [-h] | [--help] | [P] |
       [-v] [-d] [-a] [-r[G|S|A] <retries_number>] [-T[G|S|A] <timeout>] [-L <max_ttl>] [-I]
       [-l] [-c <community_name> [[community_name] ... ]] -t <max_number_of_threads>
       <base_address> <address_space_size> [[<base_address> <address_space_size>] ... ]
```

NOTE: It's important that -t option is specified as the last '-' option.

4.1 Options description

Here are described all other options (alphabetically):

```
-a            enable ARP table extraction (default = disabled)

              ARP tables are extracted using SNMP and shown in the results.
              This option can be useful for future features.

-c            community_name list to be used in SNMP requests (default = public)

              Community names defined here are used in order to test if SNMP
              is enabled on each ping responsive ip address. There is no size
              limit to this list, so a wordlist can be used in a command like
              this: hynetd -c `cat wordlist` -t . . .

-d            enable DNS inverse name look-up (default = disabled)

              With this option enabled, hostnames are retrieved using DNS inverse
              look-up queries. This way, during alias resolution phase, addresses
              resolved to the same hostname are considered aliases.

-I            set backtrace mode to ICMP (default = UDP)

              If enabled, backtrace algorithm starts its process using ICMP
              packets instead of UDP packets. It can avoid the generation
              of extra traffic when UDP packets are filtered on most routers.

-l            enable serial links heuristic method

              This method is based on IP addresses format and subnetting rules. It
              assumes that two addresses can be considered part of a serial link
              if and only if some rules are satisfied.
```

-L maximum value of ttl to use during backtrace algorithm (default = 20)

This is the maximum ttl value used to reach destination host during the backtrace algorithm. Setting this parameter to a very high value can cause performance loss. A very low value can avoid some hosts to be reached.

-P enable ping-only mode

In this mode Hynetd only scans given ranges for active addresses and writes a report in the file summary.txt

-r|-rG number of retries in scan process packet sending (default = 1)

This is the number of retries used for all the scan process operations (ping, pingRR, backtrace ...). Different values are used only during Ally algorithm and snmpTest function.

-rS number of retries in snmpTest packet sending (default = 1)

This is the number of retries used during snmpTest. It can be set to an higher value than scan process retries to maintain good performance without missing SNMP enabled routers.

-rA number of retries in Ally algorithm packet sending (default = 1)

This is the number of retries used during Ally algorithm. It can be set to an higher value to improve alias resolution results.

-T|-TG timeout value used during scan process in ms (default = 500 ms)

This is the timeout value used during all operations done in scan process (ping, pingRR, backtrace ...). Different values are used only during Ally algorithm and snmpTest function.

-TS timeout value used during snmpTest function in ms (default = 500 ms)

This is the timeout value used during snmpTest. It can be set to an higher value than scan process timeout to maintain good performance without missing SNMP enabled routers.

```
-TA          timeout value used during Ally algorithm in ms (default = 1000 ms)

             This is the timeout value used during Ally algorithm. It
             can be set to an higher value to achieve better alias resolution
             results. Values lower than 1000ms are not recommended, because
             most routers have a rate limit of 1 packet per second for
             ICMP_PORT_UNREACH replies.

-v           enable verbose mode

             If enabled, the output shown is more verbose.
```

5 Examples

In this section there are some examples that show common usages and syntax of Hynetd.

5.1 Example 1

In this example we scan 2 ranges with some options defined:

```
# hynetd -c private public -t 50 192.168.1.1 80 192.168.15.20 50
```

In this case hynetd scan addresses from **192.168.1.1** to **192.168.1.80** and **192.168.15.20** to **192.168.15.70** using 50 threads and two community names.

5.2 Example 2

Here it's shown an example that scans a range using more options defined:

```
# hynetd -l -d -c private -TS 300 -rG 0 -rS 0 -rA 0 -t 100 192.168.1.1 254
```

In this case hynetd scans the given range never doing retransmissions and waiting a timeout of 300 msec for SNMP test queries using "private" community. It also resolves DNS names and uses heuristic method to extract links during post-processing phase.

5.3 Example 3

Here it's shown an example that scans a range only to discover active adresses:

```
# hynetd -P -t 50 192.168.1.1 254
```