Mirror Your Neighbors - User manual

Jonathan Delvaux
Benjamin Vermeulen

Contents

1 Step-by-step utilisation

1. Extract the archive `MYN.zip` downloaded on `https://github.com/djo938/M.Y.N/downloads`
2. Compile the core (see section 2)
3. Open a terminal
4. Go to the MYN/core directory
5. Execute: `sudo ./start_core`
6. Open another terminal
7. Go to the MYN/core directory
8. Execute: `./command 127.0.0.1 22222 shell`
9. `M.Y.N control:>dset en1` (Capture interface selection, see section 3)

! → *en1* is an example. You can have the list of interfaces with the command **deist**

10. `M.Y.N control:>cstart tcp port 80` (Start the capture for HTTP traffic, see section 3)
11. Open Firefox
12. Install the M.Y.N extension (`MYN/misc/myn-FF-extension.xpi`)
13. Activate the extension (Tools->Run M.Y.N extension, see section 4)
14. Configure the HTTP proxy (`127.0.0.1:1200`, see section 4)
15. Open another terminal
16. Go to the MYN directory
17. Use this command to start the HTTP module:

    ```
    java -jar modules/http/pseudoProxy.jar 127.0.0.1 22223
    ```

! → *22223* is an example, the port number is given by the **cstart** command (step 10)

18. Enjoy, use for example Safari or Google Chrome and surf over the internet. You will see the result in Firefox. (Clear your cache in Safari, Google Chrome before surfing).

2 Core and command line Installation

2.1 Mac OS X

Version: The core system was tested on Mac OSX Lion 10.7.3

You need a compiler: No compiler installed on a basic Mac OSX installation, you have to install it.

If you want to known if a compiler is installed on your system, you can run the command `gcc` in a terminal window.
if the program is installed, you will see a kind of message like this :
`i686-apple-darwin11-llvm-gcc-4.2: no input files`
if the program is not installed, you will see this message :
`-bash: gcc: command not found`

A compiler is available in Xcode, you can download the last version on the apple developper center : `https://developer.apple.com/xcode/` (the program is free to download but you need to have an apple store id)

How to compile: Open a terminal window, go to `MYN/core/` directory and use the `make` command to compile the program

2.2 Linux Ubuntu

Version: The core system was tested on Ubuntu desktop 11.10 am64

Library dependencies: The core needs several libraries to compile, in order to install them, open a terminal window and write this command: `sudo apt-get install libpcap0.8-dev libreadline6 libreadline6-dev libreadline-dev readline-common`

How to compile: Open a terminal window, go to `MYN/core/` directory and use the `make` command to compile the program

3 Command Line Interface

3.1 Commands

dlist: This command allows you to enumerate all the capture devices available on the computer.

flist: This command allows you to enumerate all the capture files available on the computer. These capture files are in the folder named `capture_files` which is in the folder `src/core`. The capture files are the same file that *Wireshark* produces.

dset *capture_interface*: This command allows you to active the device to listen. The available capture device can be retrieve by using the command **dlist**.

dset *file_name*: This command allows you to load a recorded file from the list of files **flist**.

fparse: This command reads the file loaded with the command **flist**.

rstart *file_name*: This command allows you to record the stream filtered by the master filter in a file named *file_name*. The master filter can be configure with the command **mset**.

rstop: This command stops the recording started with the command **rstart**.

astop: This command stops all captures in progress.

dstop: This command allows you to deactivate the selected device.

sset: Not yet implemented.

cstart *BSP filter*: This command starts a process allowing to a module to retrieve packets on a dynamic port. If there is no argument with the command, the filter is the master filter. Otherwise, you can specify an additional filter just for this module. Several modules can be launched at the same time. BSP filters are described in the section Berkeley Packet Filter.

cstop *port_number*: This command allows to stop a capture module started with the command **cstart** by specifying the corresponding port number. (Not yet implemented)

plist: This command displays all the stream available after the master filtering. In order to save the memory of the computer, only the x last streams will be displayed (kept in memory). The x value can be configured with the command **plength**. Regarding a file parsing, all streams will be available (not just the x last streams).

plength i: This command sets the maximum number of information about streams to keep in memory.

pclear: This command allows to clean the list of streams in memory.

mset *BSP filter*: This command allows to active a filter on a the capturing interface. BSP filters are described in the section Berkeley Packet Filter.

mtest *BSP filter*: This command is a debug command useful to test the correctness of a filter before applying it. BSP filters are described in the section Berkeley Packet Filter.

4 Graphical User Interface

5 Berkeley Packet Filter

The Berkeley Packet Filter are used to filter packets on a network. This language is used in network specialized softwares like `ngrep, tcpdump, snort, ...` *. The language is defined by a set of primitives that can be composed together in order to have expressions with several primitives. When a packet match with an expression, this one is accepted by the software otherwise it is dropped.*

ip	This primitive is true for a IPv4 packet.
ip6	This primitive is true for a IPv6 packet.
src *host*	This primitive is true when the source of the IP packet is equal to *host*. This is compatible for IPv4/v6. *Host* is either an IP address or a name.
dst *host*	This primitive is the same as the previous one but it's true when the destination of the IP packet is equal to *Host*.
tcp	This primitive is true when the field *protocol* in the ip packet is set to 0x06 (TCP). This is compatible for IPv4/v6.
udp	This primitive is true when the field *protocol* in the ip packet is set to 0x11 (UDP). This is compatible for IPv4/v6.
port *i*	This primitive is true when the port in TCP or UDP is equal to *i*. This is compatible for IPv4/v6.

6 HTTP module

6.1 Firefox configuration

- First, you need to **install the last version of Firefox** on your computer (available on `http:// www.mozilla.org/`)
- The second thing to do is to **configure a proxy on Firefox** in order to deal with the core. HTTP Proxy : `127.0.0.1` port : `1200`

Figure 1: Go to the preference panel in firefox.

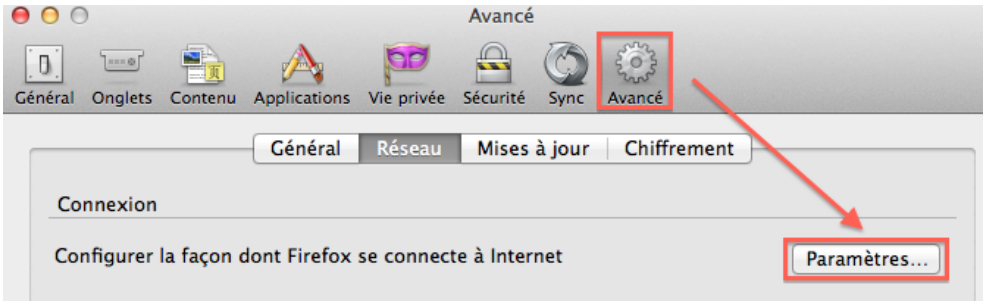

Figure 2: Go to the advanced section.

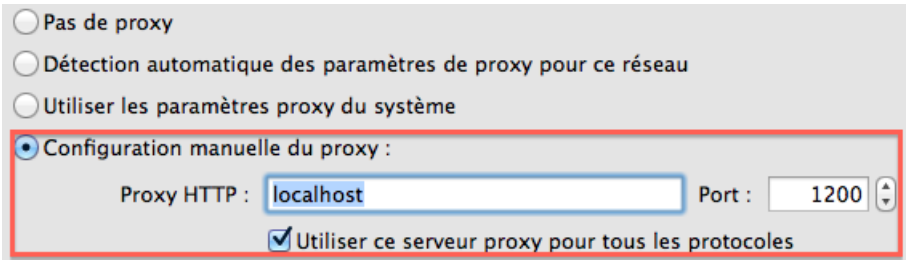

Figure 3: Enable the manual configuration of the proxy and set the address to `127.0.0.1` and the port to `1200`.

- The last thing to do is to **install the extension** M.Y.N for Firefox. This extension is located in the archive in the folder `/module/FirefoxExtension`.

  You can install the extension `myn-FF-extension.xpi` by executing this file, firefox will propose to install it for you.

6.2 Starting the HTTP module

! → **You have to start the extension in firefox before starting the HTTP module**. For starting the extension, go to the Tools menu in Firefox and click on the item *Run M.Y.N extension*.

In order to execute the HTTP module, you have to open a terminal and go to the repository where you have extracted the archive `MYN.zip`. Then you can write this command:

```
java -jar modules/http/pseudoProxy.jar <ip> <port>
```

where `<ip>` is the address of the core module and `<port>` the port number associated with a capture process. When the command (in the CLI) `cstart` *BSP Filter* is executed, a port number is available (displayed) for retrieving all the packets filtered. It's this port number which has to be used as argument `<port>`.