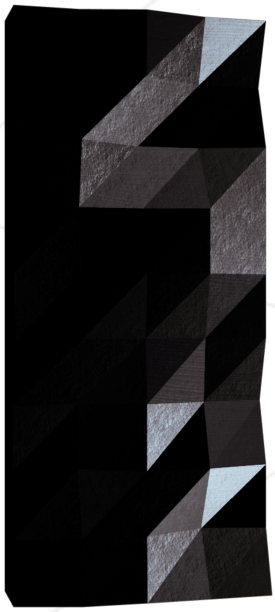

ft_nmap

Summary: This project is about recoding a part of the nmap port scanner.

Version: 3.4

Contents

Chapter I

Introduction

Nmap is a free ports scanner created by Fyodor and distributed by Insecure.org. It is conceived to detect open ports, identify hosted services and obtain information on the operating system of a distant computer. This software has become a reference for network admin because the audit of Nmap reports give indications on the network security. It is available for Windows, Mac OS X, Linux, BSD and Solaris.

Chapter II

Objectives

The goal of this project is to make you recode a part of nmap and therefore discover a new very powerful library.
You will have to use the threads in order to reduce drastically the time spent to scan the chosen ports.

```
> man nmap
```

This project implies to use mostly the PCAP library (-lpcap) and THREAD (-lpthread)

Chapter III

General Instructions

- This project will be corrected by humans only. You're allowed to organise and name your files as you see fit, but you must follow the following rules
- You must use C and submit a Makefile
- Your Makefile must compile the project and must contain the usual rules. It must recompile and re-link the program only if necessary.
- You have to handle errors carefully. In no way can your program quit in an unexpected manner (Segmentation fault, bus error, double free, etc).
- Within the mandatory part, you are allowed to use the following functions:
  - alarm
  - bind
  - connect / close
  - exit
  - fflush, fileno, fopen, fwrite, fclose
  - freeifaddrs, freeaddrinfo
  - getservbyport, getaddrinfo, getifaddrs
  - gettimeofday
  - getuid
  - htonl, htons, ntohs, ntohl
  - inet_addr
  - inet_ntoa, inet_ntop, inet_pton
  - pcap_breakloop, pcap_close, pcap_compile, pcap_dispatch
  - pcap_geterr, pcap_lookupdev, pcap_lookupnet, pcap_open_live
  - pcap_setfilter
  - perror, strerror, gai_strerror.

- poll
- pthread_create, pthread_exit, pthread_join
- pthread_mutex_init, pthread_mutex_lock, pthread_mutex_unlock
- sendto, recvfrom, recv
- setsockopt, socket
- sigaction, sigemptyset
- printf and its family.
- Your libft functions.
- You are allowed to use other functions to complete the bonus part as long as their use is justified during your defense. Be smart.

Chapter IV

Mandatory Part

Usage :

```
> ft_nmap [--help] [--ports [NUMBER/RANGED]] --ip IP_ADDRESS [--speedup [NUMBER]] [--scan [TYPE]]
```

or

```
> ft_nmap [--help] [--ports [NUMBER/RANGED]] --file FILE [--speedup [NUMBER]] [--scan [TYPE]]
```

- The executable must be named ft_nmap.
- A help menu must be available.
- You must only manage a simple IPv4 (address/hostname) as parameter for your scans.
- You must manage FQDN however you don't have to make the DNS resolution.
- It must be possible to choose the number of threads (default:0 max:250) to make the scan faster
- It must be possible to read a list of IPv4 addresses and hostname from a file (formatting is free).
- Your program must be able to run the following scans:
    - SYN, NULL, ACK, FIN, XMAS, UDP

  If the scan type is not specified then all scan types must be used.
- We must be able to run each type of scan individually, and several scans simultaneously.
- The ports to be scanned can be read as a range or individually. In the case no port is specified the scan must run with the range 1-1024.
- The number of ports scanned cannot exceed 1024.
- The resolution of service types will be requested (not the version but only the TYPE).

- The result of a scan should be as clean and clear as possible. The time frame should be easy to read.

The format used for both the arguments and the IP list file is free.

For the smarty pants (or not)... Obviously you are NOT allowed to call a real nmap.

- Here is an example of the help screen:

```
 ./ft_nmap --help
Help Screen
ft_nmap [OPTIONS]
 --help     Print this help screen
 --ports    ports to scan (eg: 1-10 or 1,2,3 or 1,5-15)
 --ip       ip addresses to scan in dot format
 --file     File name containing IP addresses to scan,
 --speedup  [250 max] number of parallel threads to use
 --scan     SYN/NULL/FIN/XMAS/ACK/UDP
```

- The following is an example of a possible result

```
> ./ft_nmap --ip x.x.x.x --speedup 70 --port 70-90 --scan SYN
Scan Configurations
Target Ip-Address : x.x.x.x
No of Ports to scan : 20
Scans to be performed : SYN
No of threads : 70
Scanning..
........
Scan took 8.32132 secs
IP address: x.x.x.x
Open ports:
Port     Service Name (if applicable) Results                  Conclusion
-------------------------------------------------------------------------------
80       http                         SYN(Open)                Open

Closed/Filtered/Unfiltered ports:
Port     Service Name (if applicable) Results                  Conclusion
-------------------------------------------------------------------------------
90       Unassigned                   SYN(Filtered)            Filtered
89       Unassigned                   SYN(Filtered)            Filtered
88       kerberos                     SYN(Filtered)            Filtered
87       link                         SYN(Filtered)            Filtered
86       Unassigned                   SYN(Filtered)            Filtered
85       Unassigned                   SYN(Filtered)            Filtered
84       Unassigned                   SYN(Filtered)            Filtered
83       Unassigned                   SYN(Filtered)            Filtered
82       Unassigned                   SYN(Filtered)            Filtered
81       Unassigned                   SYN(Filtered)            Filtered
79       finger                       SYN(Filtered)            Filtered
78       Unassigned                   SYN(Filtered)            Filtered
```

```
77      rje                     SYN(Filtered)           Filtered
76      Unassigned              SYN(Filtered)           Filtered
75      Unassigned              SYN(Filtered)           Filtered
74      Unassigned              SYN(Filtered)           Filtered
73      Unassigned              SYN(Filtered)           Filtered
72      Unassigned              SYN(Filtered)           Filtered
71      Unassigned              SYN(Filtered)           Filtered
70      gopher                  SYN(Filtered)           Filtered
```

- The following is an other example of a possible result:

```
>./ft_nmap --ip x.x.x.x --speedup 200 --port 75-85
Scan Configurations
Target Ip-Address : x.x.x.x
No of Ports to scan : 10
Scans to be performed : SYN NULL FIN XMAS ACK UDP
No of threads : 200
Scanning..
..............
Scan took 16.21338 secs
IP address: x.x.x.x
Open ports:
Port     Service Name (if applicable) Results                        Conclusion
-------------------------------------------------------------------------------------------------
80       http                    SYN(Open) NULL(Closed) FIN(Closed)
                                 XMAS(Closed) ACK(Unfiltered)
                                 UDP(Open|Filtered)                 Open

Closed/Filtered/Unfiltered ports:
Port     Service Name (if applicable) Results                        Conclusion
-------------------------------------------------------------------------------------------------
85       Unassigned              SYN(Filtered) NULL(Closed) FIN(Closed)
                                 XMAS(Closed) ACK(Unfiltered)
                                 UDP(Open|Filtered)                 Closed
84       Unassigned              SYN(Filtered) NULL(Closed) FIN(Closed)
                                 XMAS(Closed)ACK(Unfiltered)
                                 UDP(Open|Filtered)                 Closed
83       Unassigned              SYN(Filtered) NULL(Closed) FIN(Closed)
                                 XMAS(Closed) ACK(Unfiltered)
                                 UDP(Open|Filtered)                 Closed
82       Unassigned              SYN(Filtered) NULL(Closed) FIN(Closed)
                                 XMAS(Open|Filtered) ACK(Unfiltered)
                                 UDP(Open|Filtered)                 Closed
81       Unassigned              SYN(Filtered) NULL(Closed) FIN(Closed)
                                 XMAS(Closed) ACK(Unfiltered)
                                 UDP(Open|Filtered)                 Closed
79       finger                  SYN(Filtered) NULL(Closed) FIN(Closed)
                                 XMAS(Closed) ACK(Unfiltered)
                                 UDP(Open|Filtered)                 Closed
78       Unassigned              SYN(Filtered) NULL(Closed) FIN(Closed)
                                 XMAS(Closed) ACK(Unfiltered)
                                 UDP(Open|Filtered)                 Closed
77       rje                     SYN(Filtered) NULL(Open|Filtered)
                                 FIN(Closed) XMAS(Closed) ACK(Unfiltered)
                                 UDP(Open|Filtered)                 Closed
76       Unassigned              SYN(Filtered) NULL(Open|Filtered)
                                 FIN(Closed XMAS(Closed) ACK(Unfiltered)
                                 UDP(Open|Filtered)                 Closed
75       Unassigned              SYN(Filtered) NULL(Closed) FIN(Closed)
                                 XMAS(Closed) ACK(Unfiltered)
                                 UDP(Open|Filtered)                 Closed
```

Chapter V

Bonus Part

Find below a few ideas of interesting bonuses:

- DNS/Version management.
- OS detection.
- Flag to go over the IDS/Firewall.
- Being able to hide the source address.
- Additional flags...

The -v/-V flag is not a valid bonus.

The bonus part will only be assessed if the mandatory part is PERFECT. Perfect means the mandatory part has been integrally done and works without malfunctioning. If you have not passed ALL the mandatory requirements, your bonus part will not be evaluated at all.

Chapter VI

Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate to double check the names of your folders and files to ensure they are correct.

- You have to be in a VM with a Linux kernel > 3.14. Note that grading was designed on a Debian 7.0 stable.