Notes for "Nmap Network Scanning"

Anders Damsgaard Christensen

Last revision: March 9, 2011

Abstract

This document contains personal notes on the use of the Nmap Security Scanner (`http://nmap.org`), mainly based on the book *Nmap Network Scanning* by Gordon "Fyodor" Lyon. About half the document is available online[1]. This document is strictly for personal use, and redistribution is prohibited.

Contact:
`andersdc@gmail.com`
`http://cs.au.dk/~adc`

Contents

[1] `http://nmap.org/book/`

1 Introduction

Nmap is a free and open source network exploration and security auditing tool, working cross-platform altough best working on Linux-type environments. It uses raw IP packets to determine what hosts are available on the network, what services (application name and version) those hosts are running, and guesses the operational system, uptime and other characteristics. At the time of writing, the current Nmap version is 5.50.

The Nmap source code is freely available through `http://nmap.org/download.html`, along with Linux, Windows and Mac OS X binaries. The various package systems in Linux, BSD and Mac OS X systems very often contain a Nmap item.

1.1 Notation

When displaying code examples, a line beginning with `$` refers to execution in the system terminal. Field values are encapsulated by smaller-than and greater-than characters (`<` and `>`). These charaters are solely for comprehensional purposes and must *not* be typed during real use.

2 Host discovery

2.1 Specifying Nmap targets

The Nmap target(s) can consist of a range of IP adresses or hostnames. The IP adress range can be selected through CIDR-style addressing (see table 2.1). The CIDR flag is added behind a hostname or IP address. As an example, the notation `192.168.0.0/24` will scan all 256 IP's from `192.168.0.0` to `192.168.0.255`. A range of numbers can also be specified with the dash character, e.g. `192.168.0.5-12`, which will scan 8 IP addresses.

CIDR	No. of hosts	Class
/30	4	1/64 C
/29	8	1/32 C
/28	16	1/16 C
/27	32	1/8 C
/26	64	1/4 C
/25	128	1/2 C
/24	256	C
/23	512	2 C
/22	1,024	4 C
/16	65,536	B

Table 1: CIDR codes and number of IP's targeted (numbers wrap at 255—0).

When dealing with many IP addresses, it is usually more convenient to store them in a text file, and passing the file name to Nmap with `-iL <inputList>`. A number of random targets is chosen with `-iR <numberOfTargets>`. Targets within a range can be excluded with `--exclude <target>`. To check the list of

targets, specify the `-sL -n <targets>`, which prints out the target IP's, and nothing else.

2.2 Finding IP addresses

A domain name can be resolved to one or more hostnames using the `host` program:

```
Name server(s):
 $ host -t ns target.com
Address(es):
 $ host -t a target.com
AAAA record:
 $ host -t aaaa target.com
Mail servers:
 $ host -t mx target.com
SOA record:
 $ host -t soa target.com
```

These commands will yield a few hostnames. A larger list will come from zone transfer requests to the DNS server. Most DNS servers now reject these, but it is worth a try. The DNS servers will probably be revealed by the above `ns` and `soa` commands. The zone transfer request is executed with `dig`:

```
$ dig @ns01.example-dns.com -t AXFR target.com
```

Any resulting DNS results most often resolve to the target network, but may also point to third-party addresses, which are not desired to scan. Check this by performing DNS reverse-resolution and traceroute with Nmap, and finally whois with the `whois` command. Following is an example of these tests. First reverse-DNS and traceroute scan with Nmap:

```
$ nmap -PN -T4 --traceroute somehost.target.com
```

Afterwards `whois` is used on the last returned IP address from the above Nmap traceroute (largest hop no.). It uses the `http://whois.arin.net/ui` registration database for the US, and RIPE for Europe and the Middle East:

```
$ whois 207.171.166.49
```

The output will confirm wether the IP address is property of the target network. Another option is to use other web databases to find hostnames under a given domain, e.g. `http://searchdns.netcraft.com/?host`. Typing `.target.com` will bring numerous results. Alternatively Google can be searched with the string `site:target.com`.

2.3 Host discovery controls

By default, Nmap performs an initial ping scan to every target. Responsive targets are then scanned with intrusive probes such as port scans, OS detection, Nmap Scripting Engine or version detection. This behavior is configurable with the following options.

2.3.1 List scan (`-sL`)

Once one or more IP's are selected for further investigations, it is a good idea to make a reverse-DNS lookup of them. This will display the hostnames of the IP's:

```
$ nmap -sL <target(s)>
```

The list scan will not send any packets to the target hosts, and is a good measure to check wether the IP's are correct. A list scan is unobtrusive, and is unlikely to be detected. To be even more sure of not being traced, the requests can be bounced through anonymous recursive DNS servers using the `---dns-servers` option (see DNS proxying).

2.3.2 Ping scan (`-sP`)

This option will make Nmap perform nothing except the ping scan, and print out the hosts that responded to the scan. This scan mode can be expanded with the Nmap Scripting Engine (`--script`) and traceroute probing (`--traceroute`).

The ping scan is one degree more obtrusive than a list scan, but is useful when determining machine availability. It sends an ICMP echo request and a TCP ACK packet to port 80 by default. When run without sudo-priveliges, a SYN packet is sent instead, using a TCP `connect` system call to poirt 80.

The ping scan option can be combined with any of the options discussed in section 2.4, Host discovery techniques. When strict firewalls are in place between the source host running Nmap and the target network, using those advanced techniques is recommended. Otherwise hosts could be missed when the firewall drops probes or their responses.

2.3.3 Disable ping (`-PN`)

This will skip the Nmap discovery stage, and perform the requested scans and probes on *all* targets. This is usefull on target networks, which may have a firewall that blocks successive ping requests. It does, however, slow down the scan with an order of magnitude or more, but may be worth it.

If the input targets (specified in a list with `-iL` or otherwise) are known to be alive and responsive, it is unnecessary to perform a ping scan on them.

2.4 Host discovery techniques

Many firewalls have blocked ICMP ping messages (ICMP only ping probe: `-sP -PE`).

A Nmap command line options and syntax

```
Nmap 5.50 ( http://nmap.org )
Usage: nmap [Scan Type(s)] [Options] {target specification}
TARGET SPECIFICATION:
  Can pass hostnames, IP addresses, networks, etc.
  Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254
  -iL <inputfilename>: Input from list of hosts/networks
  -iR <num hosts>: Choose random targets
  --exclude <host1[,host2][,host3],...>: Exclude hosts/networks
  --excludefile <exclude_file>: Exclude list from file
HOST DISCOVERY:
  -sL: List Scan - simply list targets to scan
  -sn: Ping Scan - disable port scan
  -Pn: Treat all hosts as online -- skip host discovery
  -PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports
  -PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
  -PO[protocol list]: IP Protocol Ping
  -n/-R: Never do DNS resolution/Always resolve [default: sometimes]
  --dns-servers <serv1[,serv2],...>: Specify custom DNS servers
  --system-dns: Use OS's DNS resolver
  --traceroute: Trace hop path to each host
SCAN TECHNIQUES:
  -sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans
  -sU: UDP Scan
  -sN/sF/sX: TCP Null, FIN, and Xmas scans
  --scanflags <flags>: Customize TCP scan flags
  -sI <zombie host[:probeport]>: Idle scan
  -sY/sZ: SCTP INIT/COOKIE-ECHO scans
  -sO: IP protocol scan
  -b <FTP relay host>: FTP bounce scan
PORT SPECIFICATION AND SCAN ORDER:
  -p <port ranges>: Only scan specified ports
    Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080,S:9
  -F: Fast mode - Scan fewer ports than the default scan
  -r: Scan ports consecutively - don't randomize
  --top-ports <number>: Scan <number> most common ports
  --port-ratio <ratio>: Scan ports more common than <ratio>
SERVICE/VERSION DETECTION:
  -sV: Probe open ports to determine service/version info
  --version-intensity <level>: Set from 0 (light) to 9 (try all probes)
  --version-light: Limit to most likely probes (intensity 2)
  --version-all: Try every single probe (intensity 9)
  --version-trace: Show detailed version scan activity (for debugging)
SCRIPT SCAN:
  -sC: equivalent to --script=default
  --script=<Lua scripts>: <Lua scripts> is a comma separated list of
           directories, script-files or script-categories
  --script-args=<n1=v1,[n2=v2,...]>: provide arguments to scripts
  --script-trace: Show all data sent and received
```

```
  --script-updatedb: Update the script database.
OS DETECTION:
  -O: Enable OS detection
  --osscan-limit: Limit OS detection to promising targets
  --osscan-guess: Guess OS more aggressively
TIMING AND PERFORMANCE:
  Options which take <time> are in seconds, or append 'ms' (milliseconds),
  's' (seconds), 'm' (minutes), or 'h' (hours) to the value (e.g. 30m).
  -T<0-5>: Set timing template (higher is faster)
  --min-hostgroup/max-hostgroup <size>: Parallel host scan group sizes
  --min-parallelism/max-parallelism <numprobes>: Probe parallelization
  --min-rtt-timeout/max-rtt-timeout/initial-rtt-timeout <time>: Specifies
      probe round trip time.
  --max-retries <tries>: Caps number of port scan probe retransmissions.
  --host-timeout <time>: Give up on target after this long
  --scan-delay/--max-scan-delay <time>: Adjust delay between probes
  --min-rate <number>: Send packets no slower than <number> per second
  --max-rate <number>: Send packets no faster than <number> per second
FIREWALL/IDS EVASION AND SPOOFING:
  -f; --mtu <val>: fragment packets (optionally w/given MTU)
  -D <decoy1,decoy2[,ME],...>: Cloak a scan with decoys
  -S <IP_Address>: Spoof source address
  -e <iface>: Use specified interface
  -g/--source-port <portnum>: Use given port number
  --data-length <num>: Append random data to sent packets
  --ip-options <options>: Send packets with specified ip options
  --ttl <val>: Set IP time-to-live field
  --spoof-mac <mac address/prefix/vendor name>: Spoof your MAC address
  --badsum: Send packets with a bogus TCP/UDP/SCTP checksum
OUTPUT:
  -oN/-oX/-oS/-oG <file>: Output scan in normal, XML, s|<rIpt kIddi3,
     and Grepable format, respectively, to the given filename.
  -oA <basename>: Output in the three major formats at once
  -v: Increase verbosity level (use -vv or more for greater effect)
  -d: Increase debugging level (use -dd or more for greater effect)
  --reason: Display the reason a port is in a particular state
  --open: Only show open (or possibly open) ports
  --packet-trace: Show all packets sent and received
  --iflist: Print host interfaces and routes (for debugging)
  --log-errors: Log errors/warnings to the normal-format output file
  --append-output: Append to rather than clobber specified output files
  --resume <filename>: Resume an aborted scan
  --stylesheet <path/URL>: XSL stylesheet to transform XML output to HTML
  --webxml: Reference stylesheet from Nmap.Org for more portable XML
  --no-stylesheet: Prevent associating of XSL stylesheet w/XML output
MISC:
  -6: Enable IPv6 scanning
  -A: Enable OS detection, version detection, script scanning, and traceroute
  --datadir <dirname>: Specify custom Nmap data file location
  --send-eth/--send-ip: Send using raw ethernet frames or IP packets
```

```
  --privileged: Assume that the user is fully privileged
  --unprivileged: Assume the user lacks raw socket privileges
  -V: Print version number
  -h: Print this help summary page.
EXAMPLES:
  nmap -v -A scanme.nmap.org
  nmap -v -sn 192.168.0.0/16 10.0.0.0/8
  nmap -v -iR 10000 -Pn -p 80
SEE THE MAN PAGE (http://nmap.org/book/man.html) FOR MORE OPTIONS AND EXAMPLES
```