

Username: Palm Beach State College IP Holder **Book:** Kali Linux – Assuring Security by Penetration Testing. No part of any chapter or book may be reproduced or transmitted in any form by any means without the prior written permission for reprints and excerpts from the publisher of the book or chapter. Redistribution or other use that violates the fair use privilege under U.S. copyright laws (see 17 USC107) or that otherwise violates these Terms of Service is strictly prohibited. Violators will be prosecuted to the full extent of U.S. Federal and Massachusetts laws.

The network scanner

In this section, we will look at several tools that can be used to find open ports, fingerprint the remote operating system, and enumerate the services on the remote machine.

Service enumeration is a method that is used to find the service version that is available on a particular port on the target system. This version information is important because with this information, the penetration tester can search for security vulnerabilities that exist for that software version.

Some system administrators often change the port number, a service is listening on. For example, an SSH service may be bound to port **22** (as a convention), but a system administrator may change it to be bound to port **2222**. If the penetration tester only does a port scan to the common port of SSH, it may not find that service. The penetration tester will also have difficulties when dealing with proprietary applications running on non-standard ports. By using the service enumeration tools, these two problems can be mitigated, so there is a chance that the service can be found, regardless of the port it binds to.

Nmap

Nmap is a very comprehensive, feature- and fingerprint-rich, and widely used port scanner by all of the IT security community. It is written and maintained by Fyodor. It is a must-have tool for a penetration tester because of its quality and flexibility.

Besides being used as a port scanner, Nmap has several other capabilities as follows:

- **Host discovery:** Nmap can be used to find live hosts on the target systems. By default, Nmap will send an ICMP echo request, a TCP SYN packet to port **443**, a TCP ACK packet to port **80**, and an ICMP timestamp request to carry out the host discovery.
- **Service/version detection:** After Nmap has discovered the ports, it can further check for the service protocol, the application name, and the version number used on the target machine.
- **Operating system detection:** Nmap sends a series of packets to the remote host and examines the responses. Then, it compares these responses with its operating system fingerprint database and prints out the details if there is a match. If it is not able to determine the operating system, Nmap will provide a URL where you can submit the fingerprint to update its operating system fingerprint database. Of course, you should submit the fingerprint if you know the operating system used on the target system.
- **Network traceroute:** It is performed to determine the port and protocol that is most likely to reach the target system. Nmap traceroute starts with a high value of **Time to Live (TTL)** and decrements it until the TTL value reaches zero.
- **Nmap Scripting Engine:** With this feature, Nmap can be extended. If you want to add a check that is not included with the default Nmap, you can do so by writing the check using the Nmap scripting engine. Currently, there are checks for vulnerabilities in network services and for enumerating resources on the target system.

It is good practice to always check for new versions of Nmap. If you find the latest version of Nmap available for Kali Linux, you can update your Nmap by issuing the following commands:

```
apt-get update
apt-get install nmap
```

To start Nmap, go to the console to execute the following command:

```
nmap
```

This will display all of the Nmap options with their descriptions.

A new user to Nmap will find the available options quite overwhelming.

Fortunately, you only need one option to scan for the remote machine. That option is your target IP address or hostname if you have set up the DNS correctly. This is done with the following command:

```
nmap 192.168.56.103
```

The following is the result of the scan without any other options:

```
Nmap scan report for 192.168.56.103
Host is up (0.0046s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
```

```

21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 08:00:27:43:15:18 (Cadmus Computer Systems)

```

Nmap done: 1 IP address (1 host up) scanned in 13.49 seconds

From the preceding result, we can see that the target machine is very vulnerable to attack because it has many open ports.

Before we continue to use Nmap, let's take a look at the port states that can be identified by Nmap. There are six port states that are recognized by Nmap as follows:

- **Open:** This means that there is an application accepting a TCP connection, UDP datagram, or SCTP association.
- **Closed:** This means that although the port is accessible, there is no application listening on the port.
- **Filtered:** This means that Nmap can't determine whether the port is open or not because there is a packet-filtering device blocking the probe to reach the target.
- **Unfiltered:** This means that the port is accessible, but Nmap cannot determine whether it is open or closed.
- **Open|Filtered:** This means that Nmap is unable to determine whether a port is open or filtered. This happens when a scan to open ports doesn't give a response. It can be achieved by setting the firewall to drop packets.
- **Closed|Filtered:** This means Nmap is unable to determine whether a port is closed or filtered.

After describing the port states, we will describe several options that are commonly used during penetration testing, and after that, we will use those options in our practice.

Nmap target specification

Nmap will treat everything on the command line that isn't an option or option argument as target host specification. We suggest that you use the IP address specification instead of the hostname. By using the IP address, Nmap doesn't need to do DNS resolution first. This will speed up the port scanning process.

In the current version, Nmap supports the following IPv4 address specifications:

- A single host such as **192.168.0.1** .
- A whole network of adjacent hosts by using the CIDR notation such as **192.168.0.0/24** . This specification will include 256 IP addresses ranging from **192.168.0.0** to **192.168.0.255** .
- An octet range addressing such as **192.168.2-4,6.1** . This addressing will include four IP addresses: **192.168.2.1** ,

192.168.3.1 , 192.168.4.1 , and 192.168.6.1 .

- Multiple host specifications such as 192.168.2.1 172.168.3-5,9.1

For the IPv6 address, Nmap only supports the fully qualified IPv6 format and hostname such as fe80::a8bb:ccff:fedd:eeff%eth0 .

Besides getting the target specification from the command line, Nmap also accepts target definition from a text file by using the -iL

<inputfilename> option. This option is useful if we already have the IP addresses from another program.

Make sure that the entries in that file use the Nmap-supported target specification format. Each entry must be separated by spaces, tabs, or a new line.

The following code is a sample of that file:

```
192.168.1.1-254
192.168.2.1-254
```

Now let's scan a network of 192.168.56.0/24 . We want to see the packets sent by Nmap. To monitor the packets sent, we can use a packet capture utility such as tcpdump .

Open a console and type the following command:

```
tcpdump -nnX tcp and host 192.168.56.102
```

The 192.168.56.102 IP address belongs to our machine, which launches Nmap. You need to adjust it to your configuration.

Open another console on the same machine and type the following command:

```
nmap 192.168.56.0/24
```

In the tcpdump console, you will see the following packet:

```
22:42:12.107532 IP 192.168.56.102.49270 > 192.168.56.103.23: Flags [S],
seq 239440322, win 1024, options [mss 1460], length 0
 0x0000: 4500 002c eb7f 0000 3006 ad2e c0a8 3866 E...,....0.....8f
 0x0010: c0a8 3867 c076 0017 0e45 91c2 0000 0000 ..8g.v...E.....
 0x0020: 6002 0400 4173 0000 0204 05b4 `...As.....
```

From the preceding packet information, we know that the attacking machine sent a packet with a SYN flag set from port 49270 to the target machine port 23 (Telnet). The SYN flag is set by default if Nmap is run by the privileged user, such as root in Kali Linux.

The following screenshot shows other packets sent by the attacking machine to other machines and ports on the target network:

```
22:31:01.766788 IP 192.168.56.102.39755 > 192.168.56.101.22939: Flags [S], seq 340921615,
win 1024, options [mss 1460], length 0
 0x0000: 4500 002c 1714 0000 3506 7c9c c0a8 3866 E...,....5.|...8f
 0x0010: c0a8 3865 9b4b 599b 1452 0d0f 0000 0000 ..8e.KY..R.....
 0x0020: 6002 0400 8bc2 0000 0204 05b4 `.....
22:31:01.769945 IP 192.168.56.102.39754 > 192.168.56.100.1216: Flags [S], seq 340987150, w
in 1024, options [mss 1460], length 0
 0x0000: 4500 002c 40b8 0000 3406 53f9 c0a8 3866 E...,@...4.S...8f
 0x0010: c0a8 3864 9b4a 04c0 1453 0d0e 0000 0000 ..8d.J...S.....
 0x0020: 6002 0400 e09f 0000 0204 05b4 `.....
22:31:01.777222 IP 192.168.56.102.39755 > 192.168.56.100.22939: Flags [S], seq 340921615,
win 1024, options [mss 1460], length 0
 0x0000: 4500 002c e034 0000 3b06 ad7c c0a8 3866 E...,4...;...|..8f
 0x0010: c0a8 3864 9b4b 599b 1452 0d0f 0000 0000 ..8d.KY..R.....
 0x0020: 6002 0400 8bc3 0000 0204 05b4 `.....
22:31:01.786871 IP 192.168.56.102.39755 > 192.168.56.101.99: Flags [S], seq 340921615, win
1024, options [mss 1460], length 0
 0x0000: 4500 002c c6c1 0000 3106 d0ee c0a8 3866 E...,....1.....8f
 0x0010: c0a8 3865 9b4b 0063 1452 0d0f 0000 0000 ..8e.K.c.R.....
 0x0020: 6002 0400 e4fa 0000 0204 05b4 `.....
```

If the remote machine responds, the response packet will look like the following code:

```
22:36:19.939881 IP 192.168.56.103.1720 > 192.168.56.102.47823: Flags [R.],
seq 0, ack 1053563675, win 0, length 0
 0x0000: 4500 0028 0000 4000 4006 48b2 c0a8 3867 E..(..@.H...8g
 0x0010: c0a8 3866 06b8 bacf 0000 0000 3ecc 1b1b ..8f.....>...
 0x0020: 5014 0000 a243 0000 0000 0000 0000 P....C.....
```

Note the flag sent—it is denoted by the character **R** which is reset. It means that port **1720** in the target machine is closed. We can verify this with the previous Nmap result.

However, if the port is open, you will see the following network traffic:

```
22:42:12.108741 IP 192.168.56.103.23 > 192.168.56.102.49270: Flags [S.],
seq 1611132106, ack 239440323, win 5840, options [mss 1460], length 0
 0x0000: 4500 002c 0000 4000 4006 48ae c0a8 3867 E.,...@.H...8g
 0x0010: c0a8 3866 0017 c076 6007 ecca 0e45 91c3 ..8f...v`....E..
 0x0020: 6012 16d0 e1bf 0000 0204 05b4 0000
```

You can see that the packet in the preceding code is to acknowledge the sequence number from the previous packet displayed. This packet has an acknowledgement number of **239440323**, while the previous packet had a sequence number of **239440322**.

Nmap TCP scan options

To be able to use most of the TCP scan options, Nmap needs a privileged user (a root-level account in the Unix world or an administrator-level account in the Windows world). This is used to send and receive raw packets. By default, Nmap will use a TCP SYN scan, but if Nmap doesn't have a privileged user, it will use the TCP connect scan. The various scans used by Nmap are as follows:

- **TCP connect scan** (**-ST**): This option will complete the three-way handshake with each target port. If the connection succeeds, the port is considered open. As a result of the need to do a three-way handshake for each port, this scan type is slow and it will most likely be logged by the target. This is the default scan option used if Nmap is run by a user who doesn't have any privileges.
- **SYN scan** (**-SS**): This option is also known as **half-open** or **SYN stealth**. With this option, Nmap sends a SYN packet and then waits for a response. A SYN/ACK response means that the port is listening, while the RST/ACK response means that the port is not listening. If there is no response or an ICMP unreachable error message response, the port is considered to be filtered. This scan type can be performed quickly and because the three-way handshake is never completed, it is unobtrusive and stealthy. This is the default scan option if you run Nmap as a privileged user.
- **TCP NULL scan** (**-SN**), **FIN scan** (**-SF**), and **XMAS scan** (**-SX**): The NULL scan doesn't set any control bits. The FIN scan only sets the FIN flag bit, and the XMAS scan sets the FIN, PSH, and URG flags. If an RST packet is received as a response, the port is considered closed, while no response means that the port is open/filtered.
- **TCP Maimon scan** (**-SM**): The TCP Maimon scan was discovered by Uriel Maimon. A scan of this type will send a packet with the FIN/ACK flag bit set. BSD-derived systems will drop the packet if the port is open, and it will respond with RST if the port is closed.
- **TCP ACK scan** (**-SA**): This scan type is used to determine whether a firewall is stateful or not and which ports are filtered. A network packet of this type only sets the ACK bit. If RST is returned, it means that the target is unfiltered.
- **TCP Window scan** (**-SW**): This scan type works by examining the **TCP Window** field of the RST packet's response. An open port will have a positive **TCP Window** value, while a closed port will have a zero window value.
- **TCP Idle scan** (**-SI**): Using this technique, no packets are sent to the target by your machine, instead the scan will bounce off to a zombie host you specify. An IDS will report the zombie as the attacker.

Nmap also supports you in creating your own custom TCP scan by giving you the option of **scanflags**. The argument to that option can be numerical, such as **9** for PSH and FIN, or symbolic names. Just put together any combination of URG, ACK, PSH, RST, SYN, FIN, ECE, CWR, ALL, and NONE in any order; for example, **--scanflags URGACKPSH** will set the flags URG, ACK, and PSH.

Nmap UDP scan options

While the TCP scan has many types of scans, the UDP scan only has one type and that is the UDP scan (**-SU**). Even though the UDP scan is less reliable compared to the TCP scan, as a penetration tester you should not ignore this scan because there may be interesting services located on these UDP ports.

The biggest problem with the UDP scan is how to perform the scan quickly. A Linux kernel limits the sending of the **ICMP Port Unreachable** message to one message per second. Doing a UDP scanning of 65,536 ports to a machine will take more than 18 hours to complete.

To help mitigate this problem, there are several ways that can be used as follows:

- Running the UDP scan in parallel
- Scanning the most popular ports first
- Scanning behind the firewall
- Setting the `--host-timeout` option to skip slow hosts

These methods can help to decrease the time required for doing UDP port scans.

Let's see a scenario where we want to find which UDP ports are open on the target machine. To speed up the scanning process, we will only check for ports `53` (DNS) and `161` (SNMP). The following is the command used to do this:

```
nmap -sU 192.168.56.103 -p 53,161
```

The following is the result of this command:

```
Nmap scan report for 192.168.56.103
Host is up (0.0016s latency).
PORT      STATE SERVICE
53/udp    open  domain
161/udp   closed snmp
```

Nmap port specification

In the default configuration, Nmap will only scan the 1000 most common ports for each protocol randomly. The `nmap-services` file contains a popularity score for the selection of top ports.

To change that configuration, Nmap provides several options as follows:

- `-p port_range` : Scan only the defined ports. To scan port `1` to `1024` , the command is `-p 1-1024` . To scan port `1` to `65535` , the command is `-p-`
- `-F` (fast): This will scan only 100 common ports
- `-r` (don't randomize port): This option will set sequential port scanning (from lowest to highest)
- `--top-ports <1 or greater>` : This option will only scan the *N* highest-ratio ports found in the nmap-service file

To scan for ports `22` and `25` using the TCP NULL scan method, you can use the following command:

```
nmap -sN -p 22,25 192.168.56.103
```

The following command lines are the result:

```
Nmap scan report for 192.168.56.103
Host is up (0.00096s latency).
PORT      STATE SERVICE
22/tcp    open|filtered ssh
25/tcp    open|filtered smtp
80/tcp    open|filtered http
3306/tcp  open|filtered mysql
MAC Address: 08:00:27:43:15:18 (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 14.38 seconds
```

The following are the packet's dumped snippets:

```

23:23:38.581818 IP 192.168.56.102.61870 > 192.168.56.103.22: Flags [], win
1024, length 0
    0x0000:  4500 0028 06e4 0000 2f06 92ce c0a8 3866  E..(..../.8f
    0x0010:  c0a8 3867 f1ae 0016 dd9e bf90 0000 0000  ..8g.....
    0x0020:  5000 0400 2ad2 0000                                P...*...

23:23:38.581866 IP 192.168.56.102.61870 > 192.168.56.103.25: Flags [], win
1024, length 0
    0x0000:  4500 0028 1117 0000 3106 869b c0a8 3866  E..(....1....8f
    0x0010:  c0a8 3867 f1ae 0019 dd9e bf90 0000 0000  ..8g.....
    0x0020:  5000 0400 2acf 0000                                P...*...

23:23:39.683483 IP 192.168.56.102.61871 > 192.168.56.103.25: Flags [], win
1024, length 0
    0x0000:  4500 0028 afaf 0000 2706 f202 c0a8 3866  E..(....'....8f
    0x0010:  c0a8 3867 f1af 0019 dd9f bf91 0000 0000  ..8g.....
    0x0020:  5000 0400 2acc 0000                                P...*...

23:23:39.683731 IP 192.168.56.102.61871 > 192.168.56.103.22: Flags [], win
1024, length 0
    0x0000:  4500 0028 5488 0000 3506 3f2a c0a8 3866  E..(T...5.?*.8f
    0x0010:  c0a8 3867 f1af 0016 dd9f bf91 0000 0000  ..8g.....
    0x0020:  5000 0400 2acf 0000                                P...*...

```

From the packets displayed in the preceding code, we can see that:

- In the first and second packet, the attacking machine checks whether port **22** on the target machine is open. After a period of time, it checks port **25** on the target machine.
- In the third and fourth packet, the attacking machine checks whether port **25** on the target machine is open. After a period of time, it checks port **22** on the target machine.
- After waiting for some time, as there is still no response from the target machine, Nmap concludes that those two ports are open or filtered.

Nmap output options

The Nmap result can be saved to an external file. This option is useful if you want to process the Nmap result with other tools.

Even if you save the output to a file, Nmap still displays the result on the screen.

Nmap supports several output formats as follows:

- **Interactive output:** This is a default output format, and the result is sent to the standard output.
- **Normal output** (**-ON**): This format is similar to the interactive output, but it doesn't include the runtime information and warnings.
- **XML output** (**-OX**): This format can be converted to an HTML format, parsed by the Nmap graphical user interface, or imported to the database. We suggest you use this output format as much as you can.
- **Grepable output** (**-OG**): This format is deprecated, but it is still quite popular. Grepable output consists of comments (lines starting with a pound (**#**)) and target lines. A target line includes a combination of six labeled fields separated by tabs and followed by a colon. The fields are **Host** , **Ports** , **Protocols** , **Ignored State** , **OS** , **Seq Index** , **IP ID Seq** , and **Status** . We sometimes use this output if we want to process the Nmap output using the UNIX commands such as **grep** and **awk** .

Note

You can use the `-oA` option to save the Nmap result in three formats at once (normal, XML, and greppable).

To save a scan result to an XML file (`myscan.xml`), use the following command:

```
nmap 192.168.56.103 -oX myscan.xml
```

The following is a snippet of the XML file:

```
<?xml version="1.0"?>
<?xml-stylesheet href="file:///usr/bin/./share/nmap/nmap.xsl"
type="text/xsl"?>
<!-- Nmap 6.25 scan initiated Sat Jul 20 23:50:25 2013 as: nmap -oX
myscan.xml 192.168.56.103 -->
<nmaprun scanner="nmap" args="nmap -oX myscan.xml 192.168.56.103"
start="1374339025" startstr="Sat Jul 20 23:50:25 2013" version="6.25"
xmloutputversion="1.04">
<scaninfo type="syn" protocol="tcp" numservices="1000"
services="1,3-4,6-7,9,13,17,19-26,30,32-33,37,42-

<some port numbers are deleted for brevity>

50003,50006,50300,50389,50500,50636,50800,51103,51493,52673,52822,52848,528
69,54045,54328,55055-55056,55555,55600,56737-56738,57294,57797,58080,60020,
60443,61532,61900,62078,63331,64623,64680,65000,65129,65389"/>
<verbose level="0"/>
<debugging level="0"/>
<host starttime="1374339025" endtime="1374339038"><status state="up"
reason="arp-response" reason_ttl="0"/>
<address addr="192.168.56.103" addrtype="ipv4"/>
<address addr="08:00:27:43:15:18" addrtype="mac" vendor="Cadmus Computer
Systems"/>
```

It is easier to read the HTML file instead of the XML file, so we'll convert the XML format to HTML. You can use the `xsltproc` program to do the conversion. The following command is used to convert the XML file to an HTML file:

```
xsltproc myscan.xml -o myscan.html
```

The following is the HTML report as displayed by the Iceweasel web browser included in Kali Linux:

192.168.56.103**Address**

- 192.168.56.103 (ipv4)
- 08:00:27:43:15:18 - Cadmus Computer Systems (mac)

Ports

The 977 ports scanned but not shown below are in state: **closed**

- 977 ports replied with: **resets**

Port		State (toggle closed [0] filtered [0])	Service	Reason	Product	Version	Extra info
21	tcp	open	ftp	syn-ack			
22	tcp	open	ssh	syn-ack			
23	tcp	open	telnet	syn-ack			
25	tcp	open	smtp	syn-ack			
53	tcp	open	domain	syn-ack			
80	tcp	open	http	syn-ack			
111	tcp	open	rpcbind	syn-ack			

If you want to process the Nmap XML output to your liking, there are several programming language generic XML libraries that you can use for this purpose. Also, there are several libraries specifically developed to work with an Nmap output:

- **Perl:** Nmap-Parser (<http://search.cpan.org/dist/Nmap-Parser/>)
- **Python:** python-nmap (<http://xael.org/norman/python/python-nmap/>)
- **Ruby:** Ruby Nmap (<http://rubynmap.sourceforge.net/>)
- **PowerShell:** PowerShell script to parse nmap XML output (<http://www.sans.org/windows-security/2009/06/11/powershell-script-to-parse-nmap-xml-output>)

Nmap timing options

Nmap comes with six timing modes that you can set with options (**-T**):

- **paranoid (0)** : In this timing mode, a packet is sent every 5 minutes. The packets are sent in serial. This mode is useful to avoid IDS detection.
- **sneaky (1)** : This mode sends a packet every 15 seconds, and there are no packets sent in parallel.
- **polite (2)** : This mode sends a packet every 0.4 seconds and there is no parallel transmission.
- **normal (3)** : This mode sends multiple packets to multiple targets simultaneously. This is the default timing mode used by Nmap. It balances between time and network load.
- **aggressive (4)** : Nmap will scan a given host only for 5 minutes before moving on to the next target. Nmap will not wait more than 1.25 seconds for a response.
- **insane (5)** : In this mode, Nmap will scan a given host for only 75 seconds before moving on to the the next target. Nmap will not wait for more than 0.3 seconds for a response.

In our experience, the default timing mode usually works great unless you want to have a more stealthy or faster scan.

Nmap useful options

In this section, we will discuss several Nmap options that are quite useful when doing a penetration testing job.

Service version detection

Nmap can also be asked to check the service version when doing port scanning. This information is very useful when you do the vulnerability identification process later on.

To use this feature, give Nmap the **-sV** option.

The following is an example for this feature's usage. We want to find the software version used on port **22** :


```
nmap -sV 192.168.56.103 -p 22
```

The following is the result of this command:

```
Nmap scan report for 192.168.56.103
Host is up (0.0016s latency).
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
```

From the preceding information, we know that on port `22`, there is an SSH service using the `OpenSSH` software Version `4.7p1` and the SSH protocol is `2.0`.

Operating system detection

Nmap can also be asked to check the operating system used on the target machine. This information is very useful when you do the vulnerability identification process later on.

To use this feature, give Nmap the `-O` option.

The following is an example for this feature's usage. We want to find the operating system used on the target machine:

```
nmap -O 192.168.56.103
```

The following command lines are the result of this command:

```
Host is up (0.0037s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 08:00:27:43:15:18 (Cadmus Computer Systems)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
```

OS details: Linux 2.6.9 - 2.6.33

Network Distance: 1 hop

Based on the preceding information, we can see that the remote system is a Linux operating system using Linux kernel Version **2.6.9 - 2.6.33** .
If there are vulnerabilities on those Linux kernels, we can exploit them.

Disabling host discovery

If a host is blocking a ping request, Nmap may detect that the host is not active; so, Nmap may not perform heavy probing, such as port scanning, version detection, and operating system detection. To overcome this, Nmap has a feature for disabling host discovery. With this option, Nmap will assume that the target machine is available and will perform heavy probing against that machine.

This option is activated by using the **-Pn** option.

Aggressive scan

If you use the **-A** option, it will enable the following probe:

- Service version detection (**-sV**)
- Operating system detection (**-O**)
- Script scanning (**-sC**)
- Traceroute (**--traceroute**)

It may take some time for this scan type to finish. The following command can be used for aggressive scanning:

```
nmap -A 192.168.56.103
```

The following is the result of this command:

```
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
|_ssh-hostkey: 1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-methods: No Allow or Public header in OPTIONS response (status code 200)
|_http-title: Metasploitable2 - Linux
...
Host script results:
|_nbstat: NetBIOS name: METASPLOITABLE, NetBIOS user: <unknown>, NetBIOS
MAC: <unknown>
|_smb-os-discovery:
|   OS: Unix (Samba 3.0.20-Debian)
|   NetBIOS computer name:
|   Workgroup: WORKGROUP
|_   System time: 2013-07-21T09:20:22-04:00

TRACEROUTE
HOP RTT      ADDRESS
1   1.66 ms  192.168.56.103
```

Nmap for scanning the IPv6 target

In the previous section, we discussed that you can specify an IPv6 target in Nmap. In this section, we will discuss this in depth.

For this scenario, the following is the IPv6 address of each machine involved:

- Target machine: **fe80::a00:27ff:fe43:1518**

To scan an IPv6 target, just use the **-6** option and define the IPv6 target address. Currently, you can only specify individual IPv6 addresses. The following is a sample command to do port scanning to the IPv6 address:

```
nmap -6 fe80::a00:27ff:fe43:1518
```

The following is the result of this command:

```
Nmap scan report for fe80::a00:27ff:fe43:1518
Host is up (0.0014s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
53/tcp    open  domain
2121/tcp  open  ccproxy-ftp
5432/tcp  open  postgresql
MAC Address: 08:00:27:43:15:18 (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 0.34 seconds
```

We can see that in IPv6 testing, the number of ports open are lesser compared to the IPv4 testing. This may be caused by the services on the remote machine that do not support IPv6 yet.

The Nmap scripting engine

Although Nmap itself has already become a powerful network exploration tool, with the additional scripting engine capabilities, Nmap becomes a much more powerful tool. With the **Nmap Scripting Engine (NSE)**, users can automate various networking tasks, such as checking for new security vulnerabilities in applications, detecting application versions, or other capabilities not available in Nmap. Nmap has already included various NSE scripts in its package, but users can also write their own scripts to suit their needs.

The NSE scripts utilize the Lua programming language (<http://www.lua.org>) embedded in Nmap, and currently, the NSE scripts are categorized into the following:

- **auth** : The scripts in this category are used to find the authentication set on the target system such as using the brute force technique.
- **default** : These scripts are run by using the **-sC** or **-A** options. A script will be grouped in the default category if it satisfies the following requirements:
 - It must be fast
 - It needs to produce valuable and actionable information
 - Its output needs to be verbose and concise
 - It must be reliable
 - It should not be intrusive to the target system
 - It should divulge information to the third party
- **discovery** : These scripts are used to find the network.
- **doS** : The scripts in this category may cause **Denial of Service (DoS)** on the target system. Please use them carefully.
- **exploit** : These scripts will exploit security vulnerabilities on the target system. The penetration tester needs to have permission to run these scripts on the target system.
- **external** : These scripts may divulge information to third parties.
- **fuzzer** : These scripts are used to do fuzzing to the target system.

- **intrusive** : These scripts may crash the target system or use all of the target system resources.
- **malware** : These scripts will check for the existence of malware or backdoors on the target system.
- **safe** : These scripts are not supposed to cause a service crash, **Denial of Service (DoS)**, or exploit target system.
- **version** : These scripts are used with the version detection option (**-sV**) to carry out advanced detection for the service on the target system.
- **vuln** : These scripts are used to check for security vulnerabilities on the target system.

In Kali Linux, these Nmap scripts are located in the `/usr/share/nmap/scripts` directories, and currently, Nmap Version 6.25 included with Kali Linux contains more than 430 scripts.

There are several command-line arguments that can be used to call NSE as follows:

- **-sC** or **--script=default** : This performs scan using default scripts.
- **--script <filename> | <category> | <directories>** : This performs scan using the script defined in filename, categories, or directories.
- **--script-args <args>** : This provides script argument. An example of these arguments are username or password if you use the auth category.

To do port scanning to the host `192.168.56.103` and utilize the default script categories, we can give the following command:

```
nmap -sC 192.168.56.103
```

The following is the result snippet:

```
Nmap scan report for 192.168.56.103
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp    open  ssh
| ssh-hostkey: 1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
25/tcp    open  smtp
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000,
VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN,
| ssl-cert: Subject: commonName=ubuntu804-
base.localdomain/organizationName=OCOSA/stateOrProvinceName=There is no
such thing outside US/countryName=XX
| Not valid before: 2010-03-17T14:07:45+00:00
|_Not valid after: 2010-04-16T14:07:45+00:00
|_ssl-date: 2013-07-21T08:40:20+00:00; -4s from local time.
53/tcp    open  domain
| dns-nsid:
|_ bind.version: 9.4.2
111/tcp   open  rpcbind
| rpcinfo:
|   program version   port/proto  service
|   100000  2                111/tcp    rpcbind
|   100000  2                111/udp    rpcbind
|   100003  2,3,4           2049/tcp   nfs
```

```

| 100003 2,3,4      2049/udp  nfs
| 100005 1,2,3      35075/udp mountd
| 100005 1,2,3      59685/tcp mountd
| 100021 1,3,4      37466/tcp nlockmgr
| 100021 1,3,4      60726/udp nlockmgr
| 100024 1          36880/udp status
|_ 100024 1          38557/tcp status
3306/tcp open  mysql
| mysql-info: Protocol: 10
| Version: 5.0.51a-3ubuntu5
| Thread ID: 7
| Some Capabilities: Connect with DB, Compress, SSL, Transactions, Secure
Connection
| Status: Autocommit
|_Salt: !`BijWW-x7HCVi,<*[1-
5900/tcp open  vnc
| vnc-info:
|   Protocol version: 3.3
|   Security types:
|_   Unknown security type (33554432)
6667/tcp open  irc
| irc-info: Server: irc.Metasploitable.LAN
| Version: Unreal3.2.8.1. irc.Metasploitable.LAN
| Lservers/Lusers: 0/1
| Uptime: 0 days, 0:15:26
| Source host: 50388A6E.97684684.FFFA6D49.IP
|_Source ident: OK nmap
8180/tcp open  unknown
|_http-favicon: Apache Tomcat
|_http-methods: No Allow or Public header in OPTIONS response (status code
200)
|_http-title: Apache Tomcat/5.5
MAC Address: 08:00:27:43:15:18 (Cadmus Computer Systems)

```

Host script results:

```

|_nbstat: NetBIOS name: METASPLOITABLE, NetBIOS user: <unknown>, NetBIOS
MAC: <unknown>
| smb-os-discovery:
|   OS: Unix (Samba 3.0.20-Debian)
|   NetBIOS computer name:
|   Workgroup: WORKGROUP
|_   System time: 2013-07-21T04:40:20-04:00

```

Nmap done: 1 IP address (1 host up) scanned in 46.87 seconds

From the preceding information, you can see that now the Nmap result is more thorough. This is because it utilizes the NSE default scripts.

However, if you only want specific information on the target system, you can use the script by itself. If we want to collect information about the HTTP server, we can use several HTTP scripts in NSE, such as `http-enum` , `http-headers` , `http-methods` , and `http-php-version` using the following command:

```
nmap --script http-enum,http-headers,http-methods,http-php-version -p 80
```

192.168.56.103

The following is the result of this command:

```
Nmap scan report for 192.168.56.103
Host is up (0.0010s latency).
PORT      STATE SERVICE
80/tcp    open  http
| http-enum:
|   /tikiwiki/: Tikiwiki
|   /test/: Test page
|   /phpinfo.php: Possible information file
|   /phpMyAdmin/: phpMyAdmin
|   /doc/: Potentially interesting directory w/ listing on 'apache/2.2.8
(ubuntu) dav/2'
|   /icons/: Potentially interesting folder w/ directory listing
|_  /index/: Potentially interesting folder
| http-headers:
|   Date: Sun, 21 Jul 2013 08:45:07 GMT
|   Server: Apache/2.2.8 (Ubuntu) DAV/2
|   X-Powered-By: PHP/5.2.4-2ubuntu5.10
|   Connection: close
|   Content-Type: text/html
|
|_  (Request type: HEAD)
|_http-methods: No Allow or Public header in OPTIONS response (status code
200)
| http-php-version: Versions from logo query (less accurate): 5.1.3 -
5.1.6, 5.2.0 - 5.2.17
| Versions from credits query (more accurate): 5.2.3 - 5.2.5
|_Version from header x-powered-by: PHP/5.2.4-2ubuntu5.10
MAC Address: 08:00:27:43:15:18 (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 24.47 seconds
```

By utilizing four NSE scripts related to HTTP, we gain more information regarding the target system's web server:

- There are several interesting directories to check: `Tikiwiki` , `test` , and `phpMyAdmin`
- We have an interesting file: `phpinfo.php`
- We know the server is using PHP Version `5.2.3 -5.2.5`

After discussing Nmap, let's discuss another port scanner tool.

There is a useful NSE script called Nmap NSE Vulscan (http://www.computec.ch/mruef/software/nmap_nse_vulscan-1.0.tar.gz) that can help you to map the version information you obtain from a target machine with the vulnerability database, such as CVE (<http://cve.mitre.org/>), OSVDB (<http://www.osvdb.org/>), scip VulDB (<http://www.scip.ch/?vuldb>), SecurityTracker (<http://securitytracker.com/>), and SecurityFocus (<http://www.securityfocus.com/>).

The following screenshot shows the sample result of the CVE script:


```

PORT      STATE      SERVICE      REASON      VERSION
22/tcp    open      ssh          syn-ack     OpenSSH 5.8p1 Debian lubuntu3
(Ubuntu Linux; protocol 2.0)
| vulscan: scipvuldb - http://www.scip.ch/en/?vuldb (12 findings):
| [7775] Red Hat Linux/Fedora 6 OpenSSH glibc error() privilege escalation
| [4584] OpenSSH up to 5.7 auth-options.c information disclosure
| [4282] OpenSSH 5.x Legacy Certificate Handler buffer overflow
| [2667] OpenBSD OpenSSH up to 4.5 Separation Monitor Designfehler
| [2578] OpenBSD OpenSSH up to 4.4 Signal Handler race condition
| [1999] OpenBSD OpenSSH up to 4.2p1 scp system() Designfehler
| [1724] OpenBSD OpenSSH up to 4.2p1 GSSAPIDelegateCredentials Designfehler
| [1723] OpenBSD OpenSSH up to 4.2p1 Dynamic Port Forwarding Designfehler
| [1083] Nokia IPSO 3.x OpenSSH Designfehler
| [299] OpenBSD OpenSSH 3.7p1/3.7.1p1 PAM Handler Konfigurationsfehler
| [287] OpenBSD OpenSSH up to 3.7.1 buffer_append_space() buffer overflow
| [100] OpenSSH Client IP Restrictions weak authentication
|
| cve - http://cve.mitre.org (69 findings):
| [CVE-2012-6066] freeSSHd.exe in freeSSHd through 1.2.6 allows remote
attackers to bypass authentication via a crafted session, as demonstrated
by an OpenSSH client with modified versions of ssh.c and sshconnect2.c.
| [CVE-2012-5975] The SSH USERAUTH CHANGE REQUEST feature in SSH Tectia
Server 6.0.4 through 6.0.20, 6.1.0 through 6.1.12, 6.2.0 through 6.2.5, and
6.3.0 through 6.3.2 on UNIX and Linux, when old-style password
authentication is enabled, allows remote attackers to bypass authentication
via a crafted session involving entry of blank passwords, as demonstrated
by a root login session from a modified OpenSSH client with an added
input_userauth_passwd_changereq call in sshconnect2.c.
| [CVE-2012-5536] A certain Red Hat build of the pam_ssh_agent_auth module
on Red Hat Enterprise Linux (RHEL) 6 and Fedora Rawhide calls the glibc
error function instead of the error function in the OpenSSH codebase, which
allows local users to obtain sensitive information from process memory or
possibly gain privileges via crafted use of an application that relies on
this module, as demonstrated by su and sudo.
| [CVE-2012-0814] The auth_parse_options function in auth-options.c in sshd
in OpenSSH before 5.7 provides debug messages containing authorized_keys
command options, which allows remote authenticated users to obtain
potentially sensitive information by reading these messages, as

```

Nmap options for Firewall/IDS evasion

During penetration testing, you may encounter a system that is using firewall and IDS to protect the system. If you just use the default settings, your action may get detected or you may not get the correct result from Nmap. The following options may be used to help you evade the firewall/IDS:

- **-f** (fragment packets): This purpose of this option is to make it harder to detect the packets. By specifying this option once, Nmap will split the packet into 8 bytes or less after the IP header.
- **--mtu** : With this option, you can specify your own packet size fragmentation. The **Maximum Transmission Unit (MTU)** must be a multiple of eight or Nmap will give an error and exit.
- **-D** (decoy): By using this option, Nmap will send some of the probes from the spoofed IP addresses specified by the user. The idea is to mask the true IP address of the user in the logfiles. The user IP address is still in the logs. You can use **RND** to generate a random IP address or **RND:number** to generate the **<number>** IP address. The hosts you use for decoys should be up, or you will flood the target. Also remember that by using many decoys you can cause network congestion, so you may want to avoid that especially if you are scanning your client network.
- **--source-port <portnumber>** or **-g** (spoof source port): This option will be useful if the firewall is set up to allow all incoming traffic that comes from a specific port.
- **--data-length** : This option is used to change the default data length sent by Nmap in order to avoid being detected as Nmap scans.
- **--max-parallelism** : This option is usually set to one in order to instruct Nmap to send no more than one probe at a time to the target host.

- `--scan-delay <time>` : This option can be used to evade IDS/IPS that uses a threshold to detect port scanning activity.

You may also experiment with other Nmap options for evasion as explained in the Nmap manual (<http://nmap.org/book/man-bypass-firewalls-ids.html>).

Unicornscan

Unicornscan is an information gathering and correlation engine tool. It is useful for introducing stimulus and measuring the response from a TCP/IP device. **Unicornscan** has the following features:

- Asynchronous stateless TCP port scanning
- Asynchronous stateless TCP banner grabbing
- Asynchronous UDP port scanning
- Active and passive remote OS and application identification

Unfortunately, **Unicornscan** is not included in the default installation of Kali Linux; you need to install it from the repository by giving the following command:

```
apt-get install unicornscan
```

To start **Unicornscan**, use the console to execute the following command:

```
# unicornscan -h
```

This will display all the options with their descriptions.

The main difference between **Unicornscan** and other similar tools is that it is a very fast and scalable port scanner. From our experience, the scanning of UDP ports will take a long time to finish, especially if you want to test all the ports for a network. **Unicornscan** can help you with this problem.

In **Unicornscan**, you can define how many packets you want to send per second. The higher the **packets per second (PPS)** value, the faster the scan process; but this may cause an overload on the network, so be careful when using this capability. The default PPS is **300**.

Let's scan the target using the default options in **Unicornscan**. The following is the command and the result.

To carry out a UDP scan (`-m U`) for the ports **1-65535** on machine **192.168.56.103**, display the result immediately, and to be verbose (`-Iv`), the command is as follows:

```
# unicornscan -m U -Iv 192.168.56.103:1-65535
```

The following is the reply from **Unicornscan** :

```
adding 192.168.56.103/32 mode `UDPscan' ports `1-65535' pps 300
using interface(s) eth0
scanning 1.00e+00 total hosts with 6.55e+04 total packets, should take a
little longer than 3 Minutes, 45 Seconds
```

From the preceding information, we know that by using the default PPS, this scan will take more than 3 minutes. To speed up the scanning process, let's change the packet sending rate to 10,000 (`-r 10000`):

```
unicornscan -m U -Iv 192.168.56.103/24:1-65535 -r 10000
```

The following is the response from **Unicornscan** :

```
adding 192.168.56.103/32 mode `UDPscan' ports `1-65535' pps 10000
using interface(s) eth0
scanning 1.00e+00 total hosts with 6.55e+04 total packets, should take a
little longer than 13 Seconds
```


The scanning is much faster after we change the packet sending rate. Note that you may only use this rate in a fast network, if not you don't overwhelm the network with your UDP packets.

The following is the scan result:

```

UDP open 192.168.56.103:137  ttl 64
UDP open 192.168.56.103:53   ttl 64
UDP open 192.168.56.103:41250 ttl 64
UDP open 192.168.56.103:2049  ttl 64
UDP open 192.168.56.103:111   ttl 64
sender statistics 7586.6 pps with 65544 packets sent total
listener statistics 14 packets recieved 0 packets dropped and 0 interface drops
UDP open          domain[ 53]      from 192.168.56.103  ttl 64
UDP open          sunrpc[ 111]     from 192.168.56.103  ttl 64
UDP open          netbios-ns[ 137]  from 192.168.56.103  ttl 64
UDP open          shilp[ 2049]     from 192.168.56.103  ttl 64
UDP open          unknown[41250]    from 192.168.56.103  ttl 64

```

Zenmap

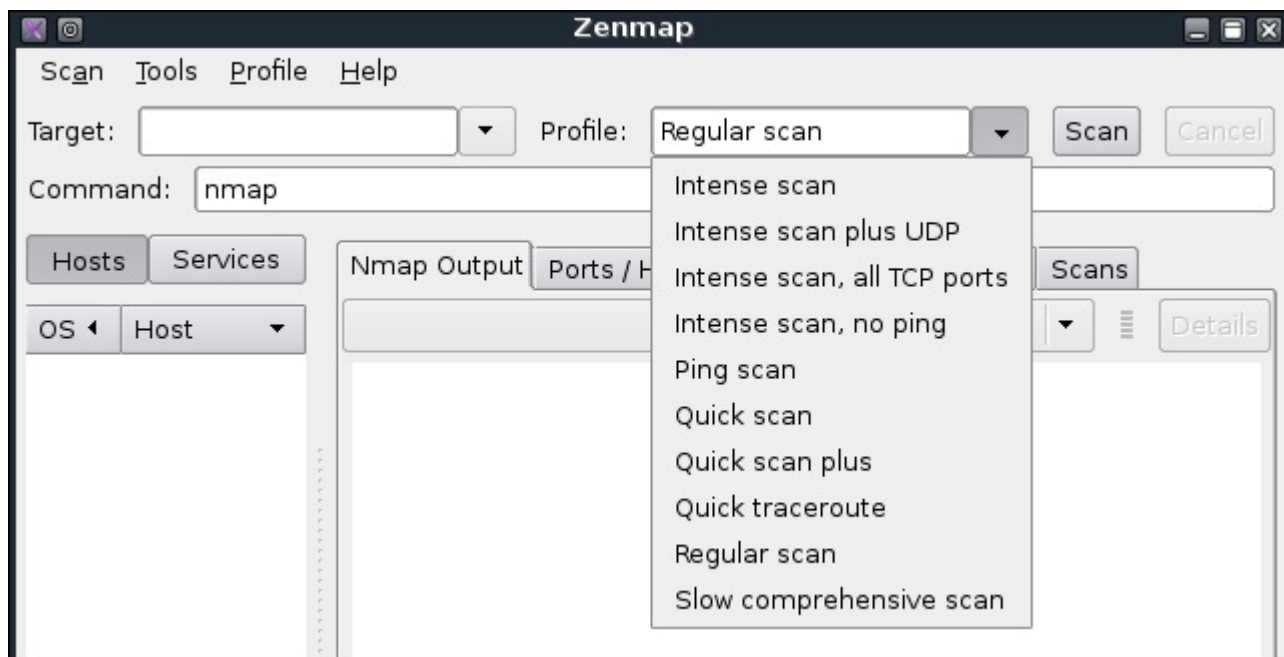
Zenmap is the graphical interface of Nmap. The advantages of Zenmap compared to Nmap are as follows:

- Zenmap is interactive; it arranges the scan results in a convenient way. It can even draw a topological map of the discovered network.
- Zenmap can do a comparison between two scans.
- Zenmap keeps a track of the scan results.
- To run the same scan configuration more than once, the penetration tester can use a Zenmap profile.
- Zenmap will always display the command that is run, so the penetration tester can verify that command.

To start Zenmap, navigate to **Kali Linux | Information Gathering | Network Scanners | Zenmap**, or use the console to execute the following command:

```
#zenmap
```

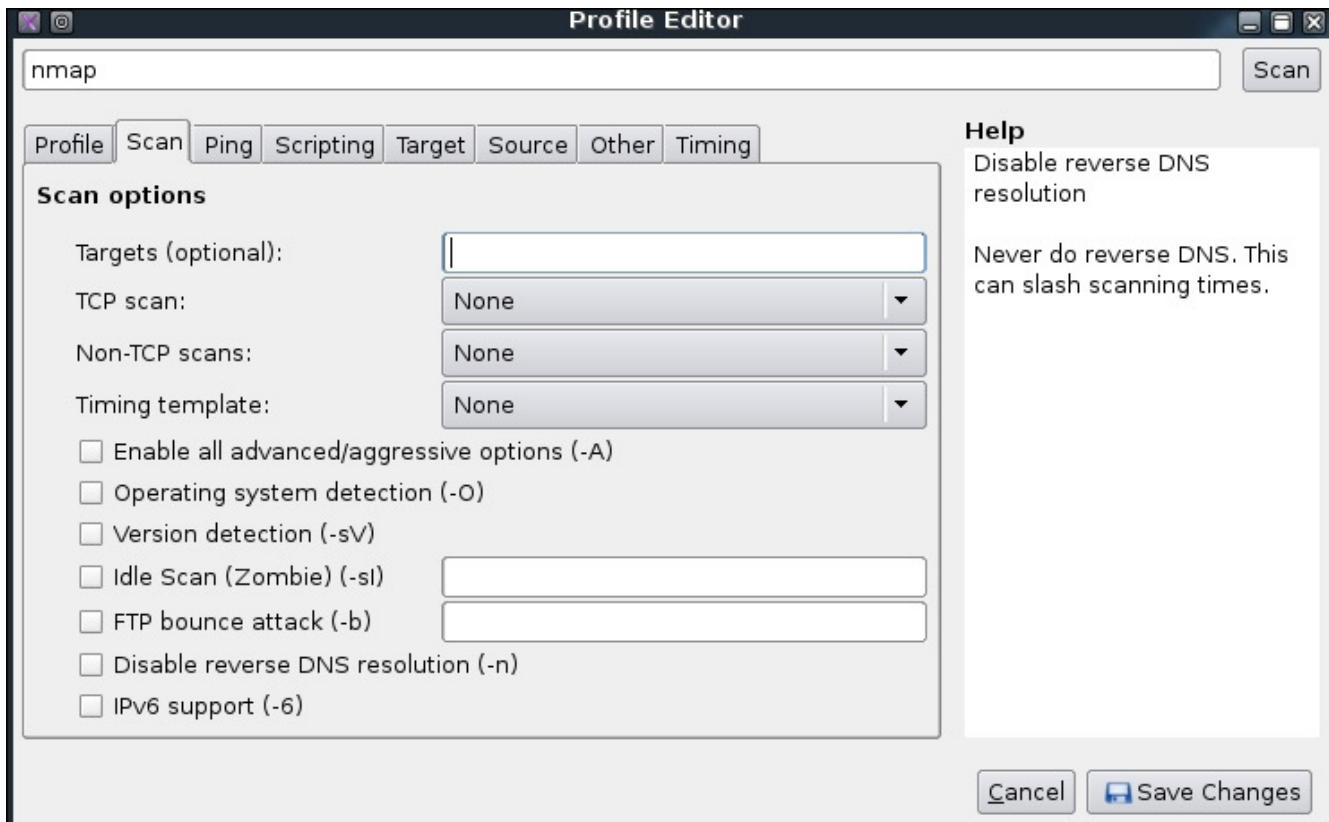
This will display the main Zenmap window. Zenmap comes with 10 profiles that can be chosen. To find which command options are used on each profile, just click on **Profile** and the command options will be displayed in the **Command:** box as shown in the following screenshot:



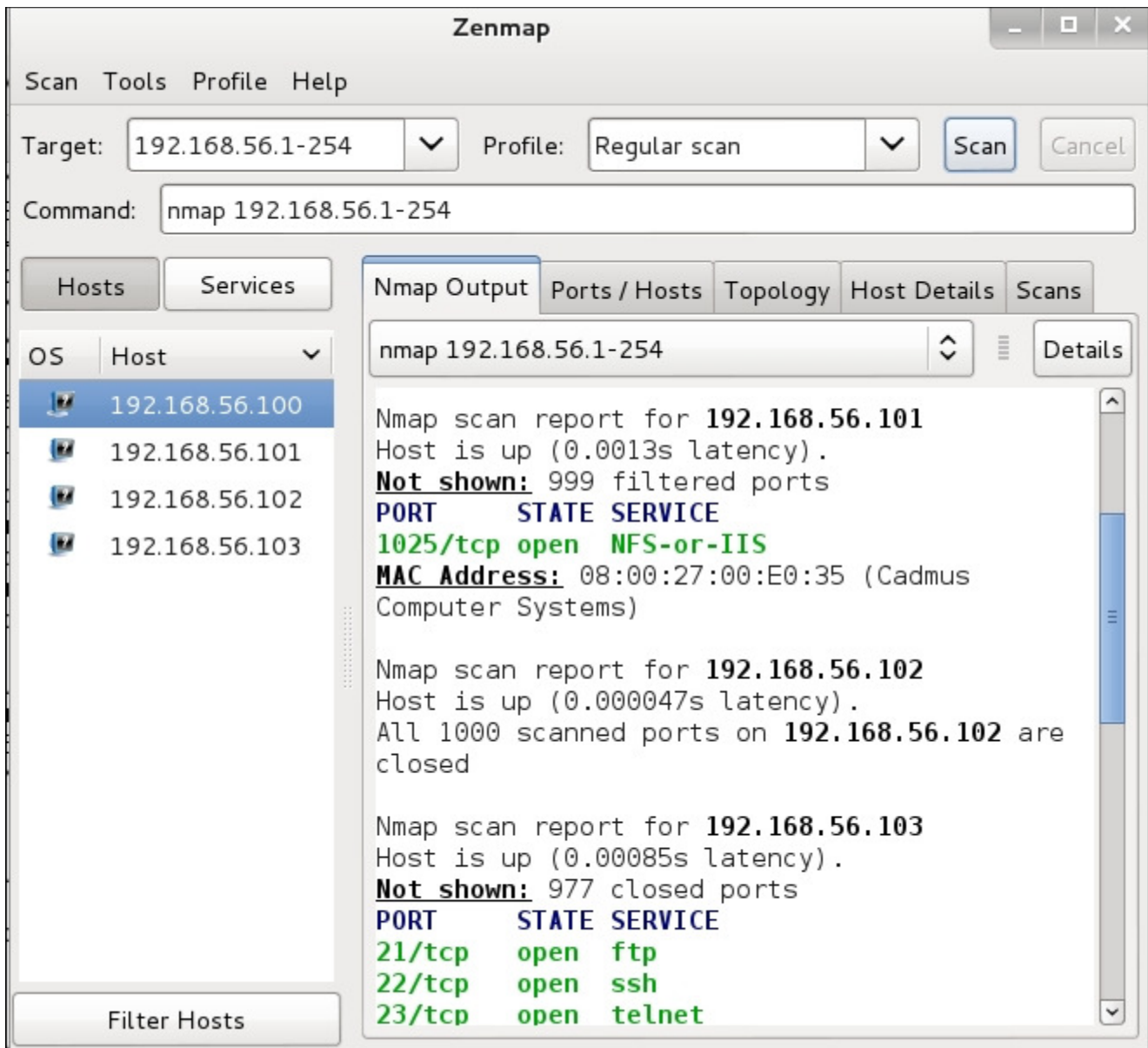
If the provided profiles are not suitable for our needs, we can create our own profile by creating a new profile or editing the existing ones. These tasks can be found under the **Profile** menu.

To create a new profile, select the menu **New Profile** or **Command** or you can press the keys *Ctrl + P*. To edit an existing profile, select the **Edit Selected Profile** menu or press *Ctrl + E*.

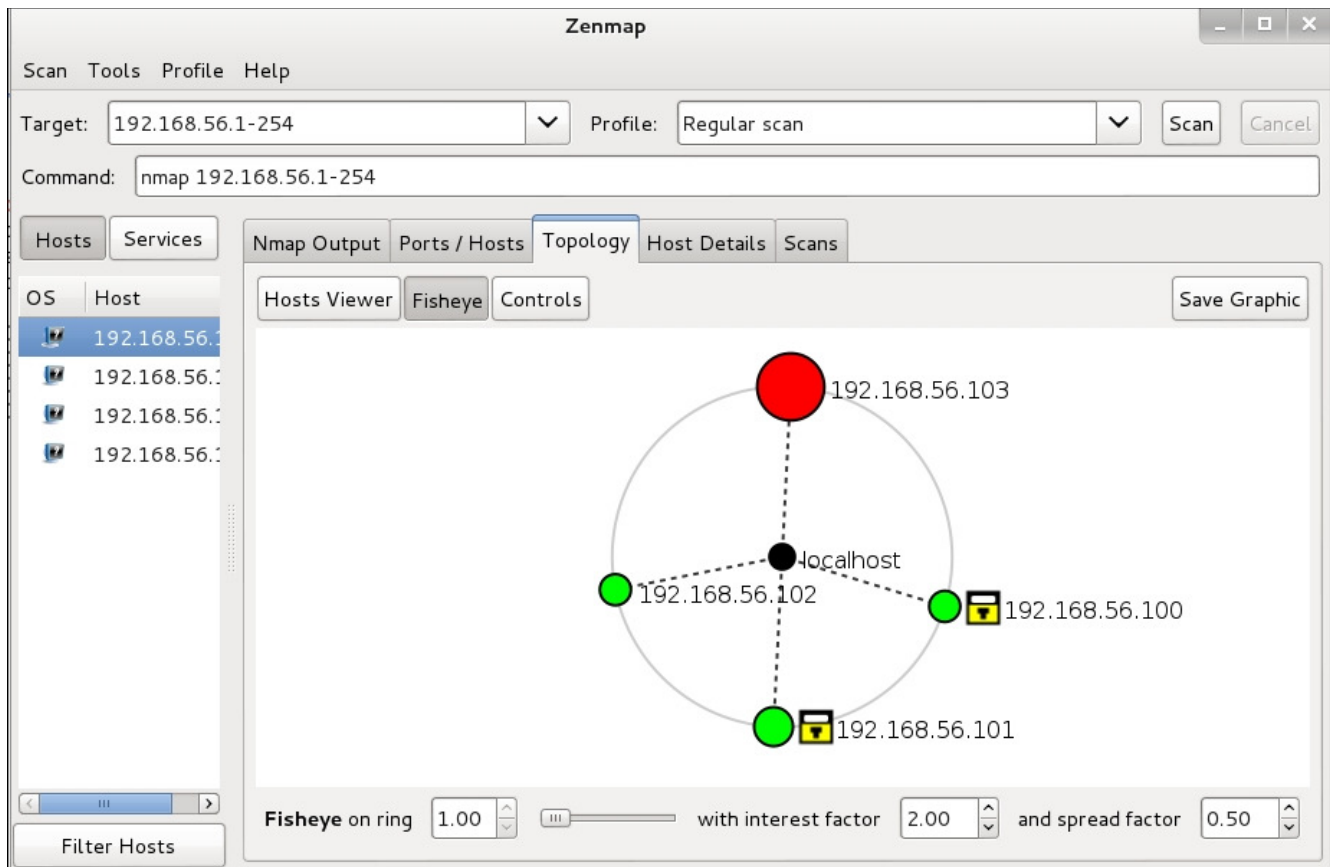
Select each tab (**Profile**, **Scan**, **Ping**, **Scripting**, **Target**, **Source**, **Other**, and **Timing**) and configure it according to your needs. If you have finished configuring the profile, save the profile by clicking on the **Save Changes** button as shown in the following screenshot:



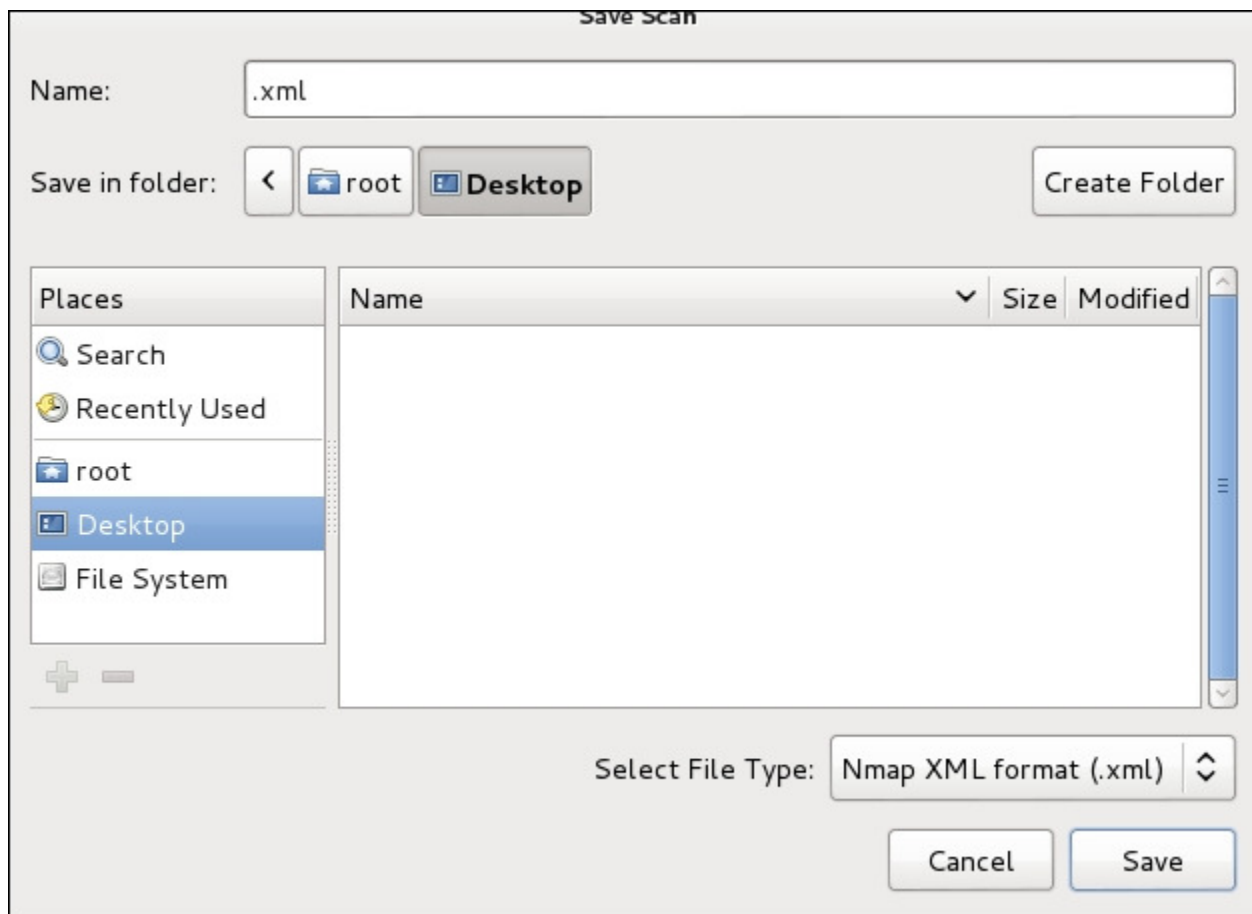
Let's scan the host **192.168.56.1-254** using the **Regular scan** profile as shown in the following screenshot:



If you want to see the network topology, click on the **Topology** tab and you will be able to see the details as shown in the following screenshot:



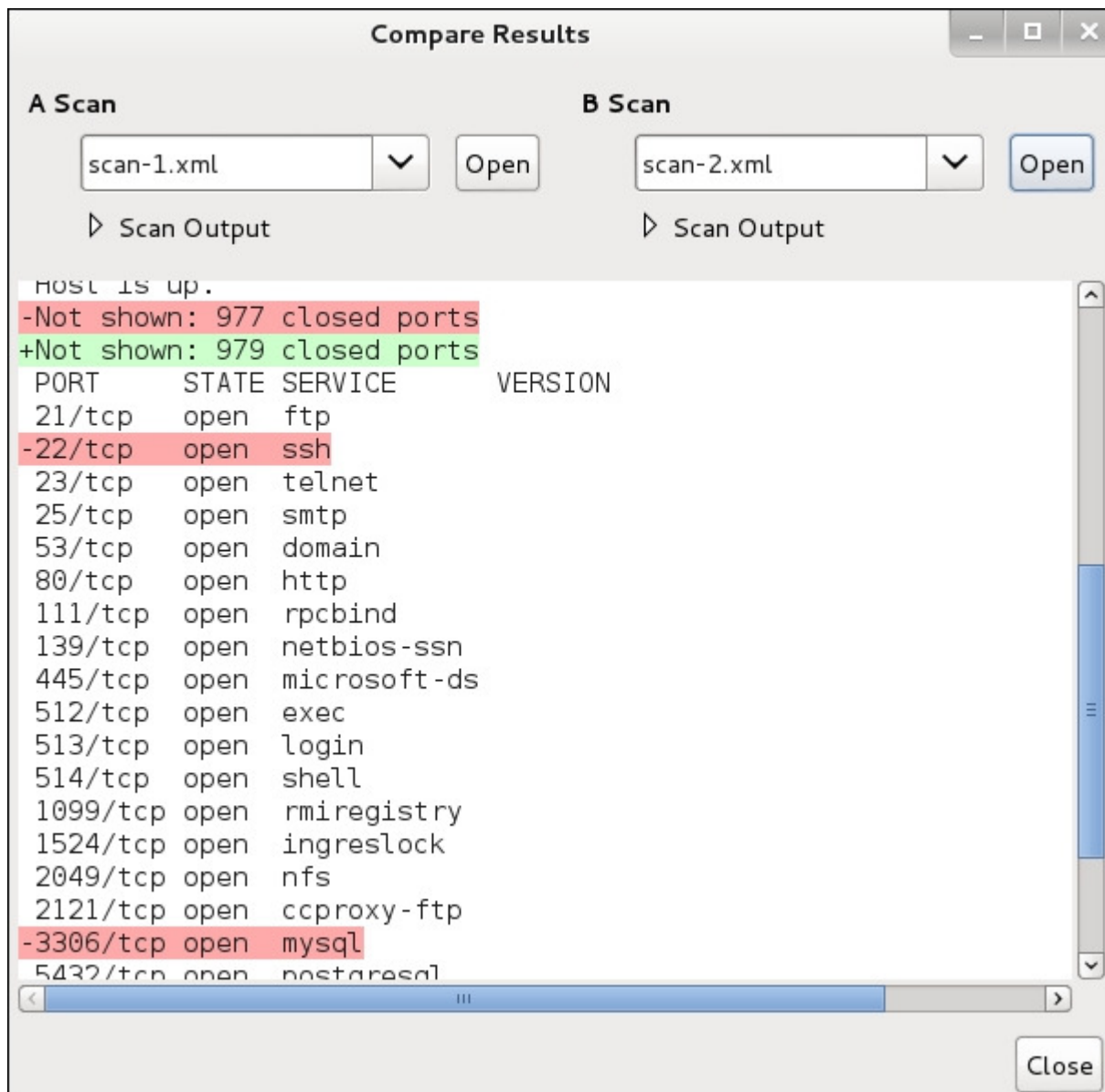
To save the Zenmap result, go to the **Scan** menu and choose **Save Scan**. Zenmap will then ask you where you want to save the result. The default format is XML as shown in the following screenshot:



To find the differences between the scans, perform the first scan and then save the result. Then, make changes to the scan targets. Next, do the second scan

and save the result. Later, compare the scan results by going to the **Tools** menu and select **Compare Results**.

For **A Scan**, you can select the XML file of the first scan result by clicking on the **Open** button, while for **B Scan**, you can select the XML file of the second scan result as shown in the following screenshot:



The - character denotes that this line is removed in the **B Scan** result, while the + character means that this line is added in the **B Scan** result.

We noticed that the SSH and MySQL ports are not open anymore in the second scan and the number of closed ports has increased from **977** to **979** to adjust with the number of closing ports during the second port scanning process.

Amap

Amap is a tool that can be used to check the application running on a specific port. **Amap** works by sending a trigger packet to the port and comparing the response with its database. It will print the application information if the application's response matches the database information.

In Kali Linux, the **Amap** trigger file is located in `/etc/apmap/appdefs.trig`, whereas the response file is available in `/etc/amap/appdefs.resp`.

To start **Amap**, go to the console and execute the following command:

```
amap
```

This will display a simple usage instruction and example on your screen.

For our exercise, we will analyze the application that runs on the target system's port **22**. We will use the **-b** and **-q** options to get banner information without reporting the closed or unidentified ports as given in the following command:

```
amap -bq 192.168.56.103 22
```

The following is the result of this command:

```
Protocol on 192.168.56.103:22/tcp matches ssh - banner: SSH-2.0-
OpenSSH_4.7p1 Debian-8ubuntu1\n
Protocol on 192.168.56.103:22/tcp matches ssh-openssh - banner: SSH-2.0-
OpenSSH_4.7p1 Debian-8ubuntu1\n
```

Using **Amap** , we can identify the application used on a specific port and the version information too.

To identify more than one port, define the ports on the command line separated by a space as follows:

```
amap -bq 192.168.56.103 80 3306
```

The following is the result of this command:

```
Protocol on 192.168.56.103:3306/tcp matches mysql - banner: >\n5.0.51a-
3ubuntu5/?,\`~yel,nd,M~Ti3ap/5Bad handshake
Protocol on 192.168.56.103:22/tcp matches ssh - banner: SSH-2.0-
OpenSSH_4.7p1 Debian-8ubuntu1\n
Protocol on 192.168.56.103:22/tcp matches ssh-openssh - banner: SSH-2.0-
OpenSSH_4.7p1 Debian-8ubuntu1\n
```

Amap is able to identify the service that is running on port **3306** , but it gives several matches when identifying the service running on port **22** .

Amap is useful if you want a quick way to find out the application service information.