MIS 311

INFORMATION SECURITY AND DESIGN

TARGET SPECIFICATION WITH NMAP

SAFA BURAK GÜRLEYEN

FATMA ATEŞ
18030411060

MAHA EL-SAYED
19030441009

JUNE,2021

What is TCP?

TCP stands for Transmission Control Protocol a communications standard that enables application programs and computing devices to exchange messages over a network. It is designed to send packets across the internet and ensure the successful delivery of data and messages over networks.

TCP is one of the basic standards that define the rules of the internet and is included within the standards defined by the Internet Engineering Task Force (IETF). It is one of the most commonly used protocols within digital network communications and ensures end-to-end data delivery.

TCP organizes data so that it can be transmitted between a server and a client. It guarantees the integrity of the data being communicated over a network. Before it transmits data, TCP establishes a connection between a source and its destination, which it ensures remains live until communication begins. It then breaks large amounts of data into smaller packets, while ensuring data integrity is in place throughout the process.

As a result, high-level protocols that need to transmit data all use TCP Protocol. Examples include peer-to-peer sharing methods like File Transfer Protocol (FTP), Secure Shell (SSH), and Telnet. It is also used to send and receive email through Internet Message Access Protocol (IMAP), Post Office Protocol (POP), and Simple Mail Transfer Protocol (SMTP), and for web access through the Hypertext Transfer Protocol (HTTP).

An alternative to TCP is the User Datagram Protocol (UDP), which is used to establish low-latency connections between applications and decrrease transmissions time. TCP can be an expensive network tool as it includes absent or corrupted packets and protects data delivery with controls like acknowledgments, connection startup, and flow control.

UDP does not provide error connection or packet sequencing nor does it signal a destination before it delivers data, which makes it less reliable but less expensive. As such, it is a good option for time-sensitive situations, such as Domain Name System (DNS) lookup, Voice over Internet Protocol (VoIP), and streaming media.

What is IP?

The Internet Protocol (IP) is the method for sending data from one device to another across the internet. Every device has an IP address that uniquely identifies it and enables it to communicate with and exchange data with other devices connected to the internet.

IP is responsible for defining how applications and devices exchange packets of data with each other. It is the principal communications protocol responsible for the formats and rules for exchanging data and messages between computers on a single network or several internet-connected networks. It does this through the Internet Protocol Suite (TCP/IP), a group of communications protocols that are split into four abstraction layers.

IP is the main protocol within the internet layer of the TCP/IP. Its main purpose is to deliver data packets between the source application or device and the destination using methods and structures that place tags, such as address information, within data packets.

TCP vs. IP: What is the Difference?

TCP and IP are separate protocols that work together to ensure data is delivered to its intended destination within a network. IP obtains and defines the address—the IP address—of the application or device the data must be sent to. TCP is then responsible for transporting and routing data through the network architecture and ensuring it gets delivered to the destination application or device that IP has defined.

In other words, the IP address is akin to a phone number assigned to a smartphone. TCP is the computer networking version of the technology used to make the smartphone ring and enable its user to talk to the person who called them. The two protocols are frequently used together and rely on each other for data to have a destination and safely reach it, which is why the process is regularly referred to as TCP/IP.

IDS \ IPS

Intrusion detection systems are one of the key security components, including methods that enable monitoring of activity on the network and analysis of traffic to detect possible attacks, breaches and threats. Intrusion prevention systems are network security systems that cover the detection and Prevention of attacks. Today, networks have a complex structure, other networks, especially many access points connected to the internet, cyber attacks are diversifying and growing day by day, and at the same time, with these systems now protected by encryption or just a complex network firewall. This made it impossible to detect attacks in real time.

IDS / IPS systems have functions such as frequently monitoring the network, identifying potential threats and keeping event records (logs) about them, stopping attacks, and reporting to security administrators. In some cases, these systems can be used to expose weaknesses in institutions ' security policies. IDS / IPS can also detect attackers ' network-related information collection activities and stop attackers at this early stage.

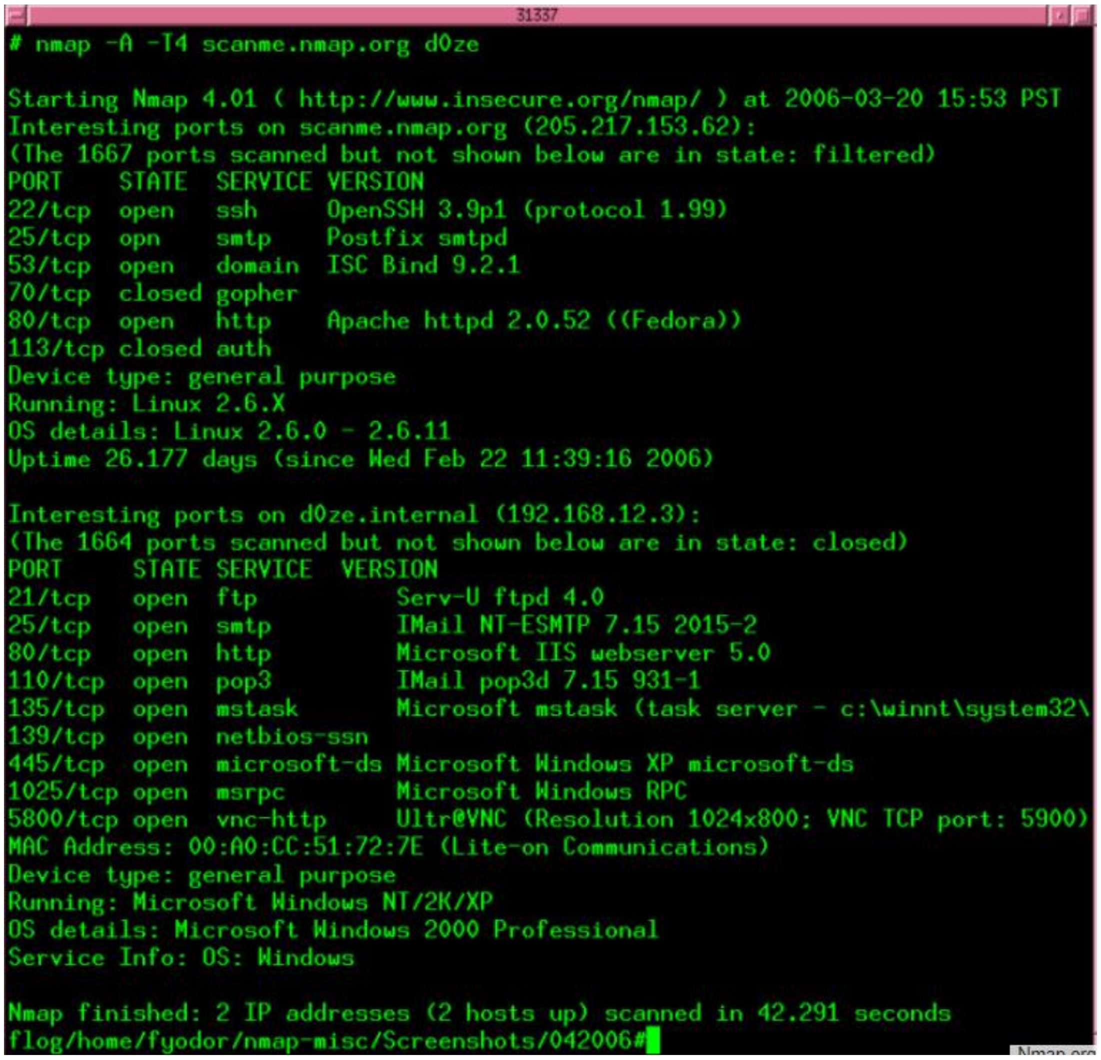
What is Nmap?

Nmap , short for Network Mapper, is a free, open-source tool for vulnerability scanning and network discovery. Network administrators use Nmap to identify what devices are running on their systems, discovering hosts that are available and the services they offer, finding open ports Nmap and detecting security risks.

Nmap can be used to monitor single hosts as well as vast networks that encompass hundreds of thousands of devices and multitudes of subnets.

Though Nmap has evolved over the years and is extremely flexible, at heart it's a port-scan tool, gathering information by sending raw packets to system ports. It listens for responses and determines whether ports are open, closed or filtered in some way by, for example, a firewall. Other terms used for port scanning include port discovery or enumeration.

Nmap is a powerful network security tool written by Gordon Lyon. It was released almost 20 years ago (in 1997) and has since become the de facto standard for network mapping and port scanning, allowing network administrators to discover hosts and services on a computer network, and create a map of the network.



```
# nmap -A -T4 scanme.nmap.org d0ze

Starting Nmap 4.01 ( http://www.insecure.org/nmap/ ) at 2006-03-20 15:53 PST
Interesting ports on scanme.nmap.org (205.217.153.62):
(The 1667 ports scanned but not shown below are in state: filtered)
PORT    STATE  SERVICE VERSION
22/tcp  open   ssh     OpenSSH 3.9p1 (protocol 1.99)
25/tcp  opn    smtp    Postfix smtpd
53/tcp  open   domain  ISC Bind 9.2.1
70/tcp  closed gopher
80/tcp  open   http    Apache httpd 2.0.52 ((Fedora))
113/tcp closed auth
Device type: general purpose
Running: Linux 2.6.X
OS details: Linux 2.6.0 - 2.6.11
Uptime 26.177 days (since Wed Feb 22 11:39:16 2006)

Interesting ports on d0ze.internal (192.168.12.3):
(The 1664 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE      VERSION
21/tcp   open  ftp          Serv-U ftpd 4.0
25/tcp   open  smtp         IMail NT-ESMTP 7.15 2015-2
80/tcp   open  http         Microsoft IIS webserver 5.0
110/tcp  open  pop3         IMail pop3d 7.15 931-1
135/tcp  open  mstask       Microsoft mstask (task server - c:\winnt\system32\
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds Microsoft Windows XP microsoft-ds
1025/tcp open  msrpc        Microsoft Windows RPC
5800/tcp open  vnc-http     Ultr@VNC (Resolution 1024x800; VNC TCP port: 5900)
MAC Address: 00:A0:CC:51:72:7E (Lite-on Communications)
Device type: general purpose
Running: Microsoft Windows NT/2K/XP
OS details: Microsoft Windows 2000 Professional
Service Info: OS: Windows

Nmap finished: 2 IP addresses (2 hosts up) scanned in 42.291 seconds
flog/home/fyodor/nmap-misc/Screenshots/042006#
```

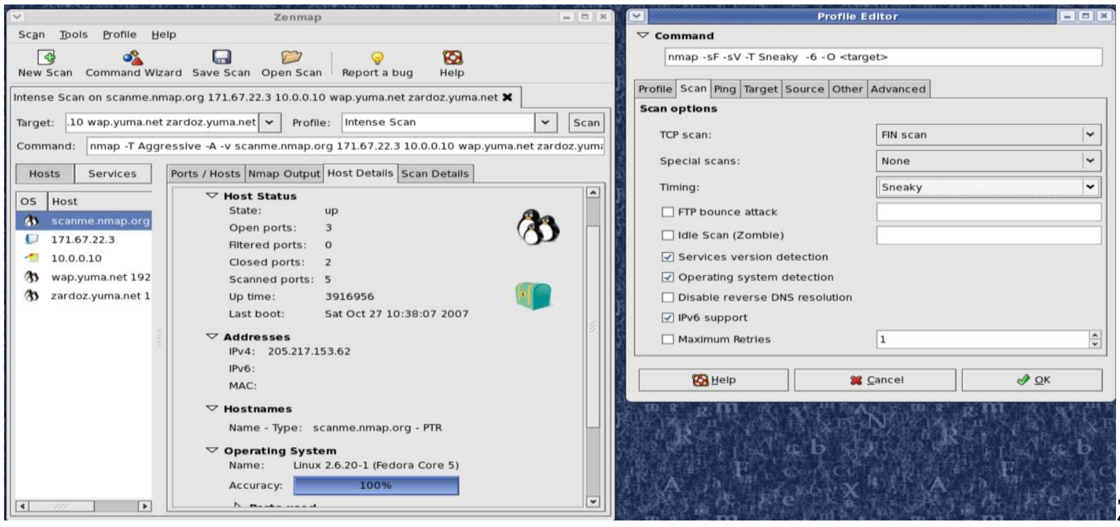
Although usually used for port scanning, Nmap offers many additional features:

- host discovery.
- operating system detection.
- service version detection.
- network information about targets, such as DNS names, device types, and MAC addresses.
- ability to scan for well-known vulnerabilities.

How to use Nmap

There is a wide range of network monitoring utilities as well as open source vulnerability scanners available to network administrators and security auditors. What makes Nmap stand out as the tool IT and network managers need to know is its flexibility and power. While the basis of Nmap's functionality is port scanning, it allows for a variety of related capabilities including:

- Network mapping: Nmap can identify the devices on a network (also called host discovery), including servers, routers and switches, and how they're physically connected.
- OS detection: Nmap can detect the operating systems running on network devices (also called OS fingerprinting), providing the vendor name, the underlying operating system, the version of the software and even an estimate of devices' uptime.
- Service discovery: Nmap can not only identify hosts on the network, but whether they're acting as mail, web or name servers, and the particular applications and versions of the related software they're running.
- Security auditing: Figuring out what versions of operating systems and applications are running on network hosts lets network managers determine their vulnerability to specific flaws. If a network admin receives an alert about a vulnerability in a particular version of an application, for example, she can scan her network to identify whether that software version is running on the network and take steps to patch or update the relevant hosts. Scripts can also automate tasks such as detecting specific vulnerabilities.



Intense Scan on scanme.nmap.org 171.67.22.3 10.0.0.10 wap.yuma.net zardoz.yuma.net

Target: .10 wap.yuma.net zardoz.yuma.net   Profile: Intense Scan

Command: nmap -T Aggressive -A -v scanme.nmap.org 171.67.22.3 10.0.0.10 wap.yuma.net zardoz.yuma

OS	Host
	scanme.nmap.org
	171.67.22.3
	10.0.0.10
	wap.yuma.net 192
	zardoz.yuma.net 1

Host Status

```
State:           up
Open ports:      3
Filtered ports:  0
Closed ports:    2
Scanned ports:   5
Up time:         3916956
Last boot:       Sat Oct 27 10:38:07 2007
```

Addresses

```
IPv4:  205.217.153.62
IPv6:
MAC:
```

Hostnames

```
Name - Type:  scanme.nmap.org - PTR
```

Operating System

```
Name:       Linux 2.6.20-1 (Fedora Core 5)
Accuracy:   100%
```

Command

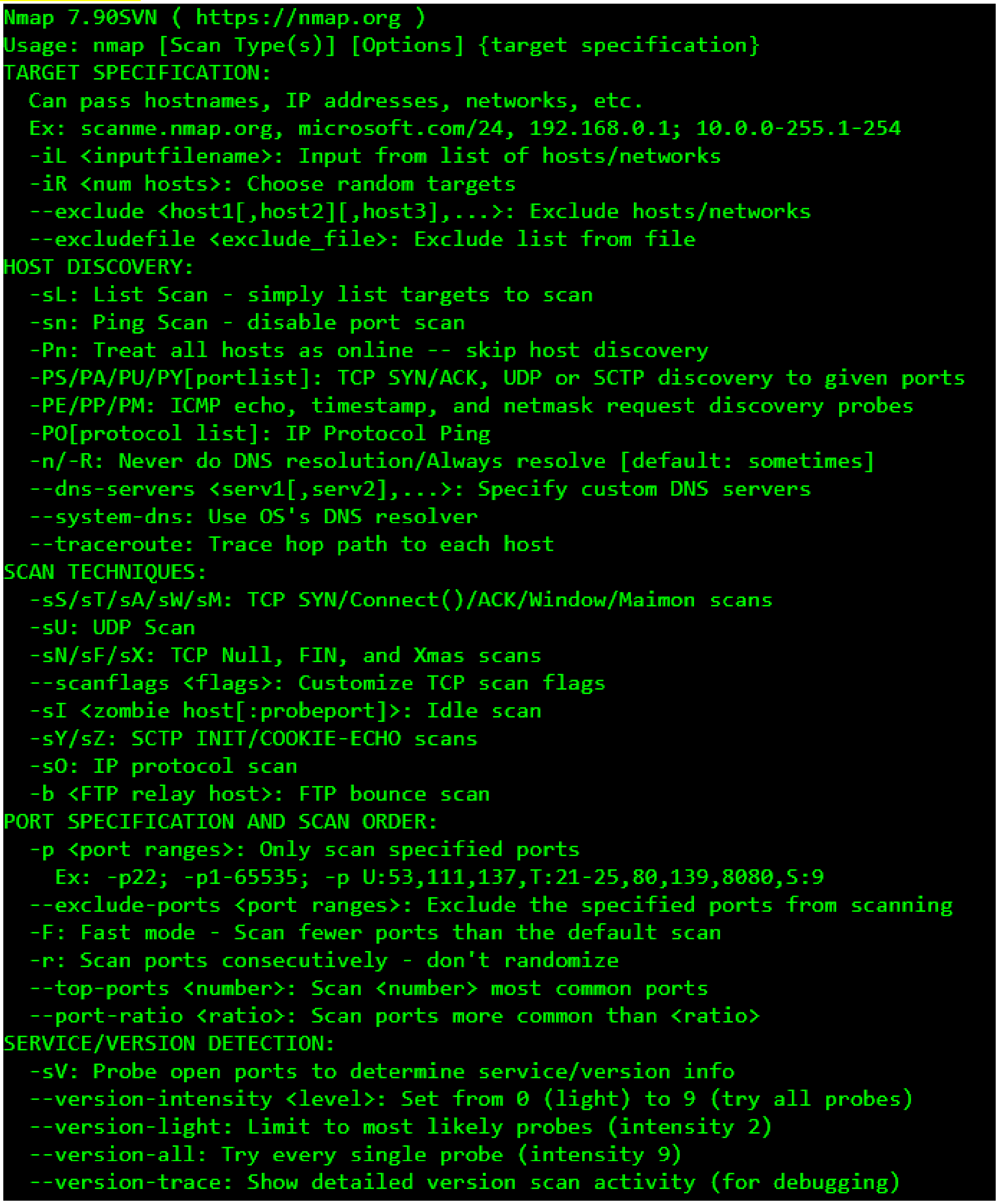
```
nmap -sF -sV -T Sneaky  -6 -O <target>
```

Scan options

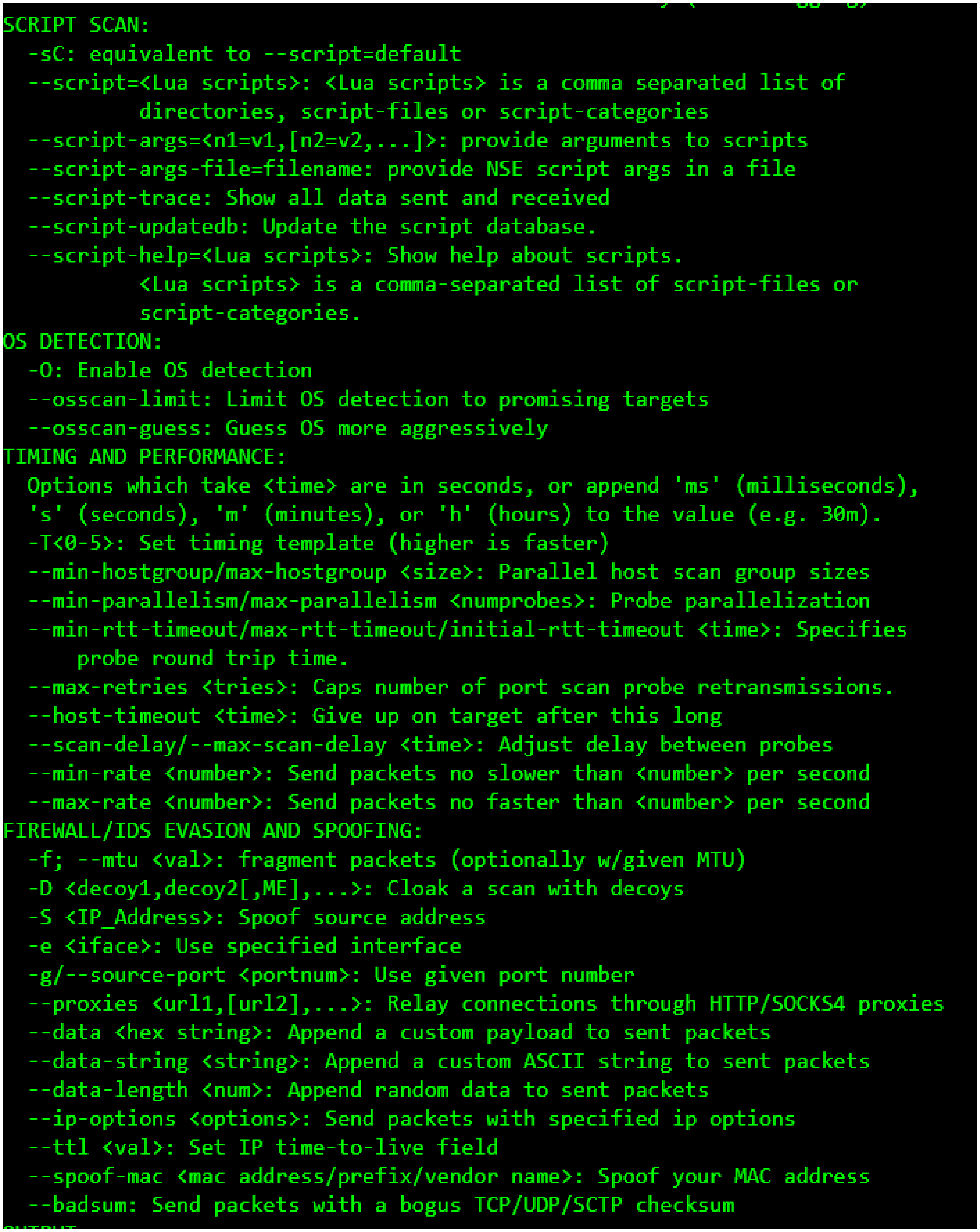
TCP scan:	FIN scan
Special scans:	None
Timing:	Sneaky
☐ FTP bounce attack	
☐ Idle Scan (Zombie)	
☑ Services version detection	
☑ Operating system detection	
☐ Disable reverse DNS resolution	
☑ IPv6 support	
☐ Maximum Retries	1

Ze
nmap is the graphical user interface for Nmap

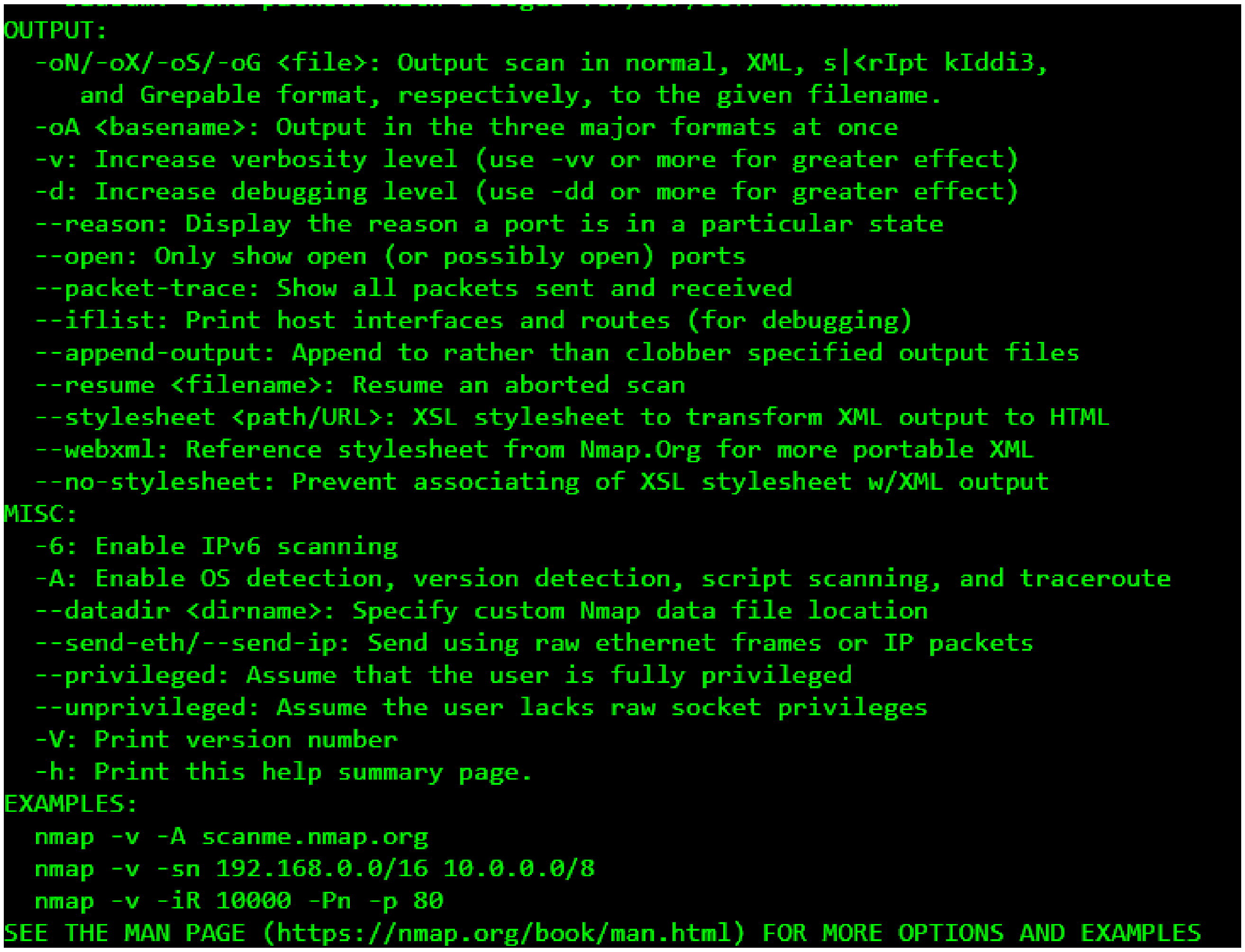
OPTION SUMMARY



```
Nmap 7.90SVN ( https://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
  --exclude-ports <port ranges>: Exclude the specified ports from scanning
  -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-args-file=filename: provide NSE script args in a file
  --script-trace: Show all data sent and received
  --script-updatedb: Update the script database.
  --script-help=<Lua scripts>: Show help about scripts.
           <Lua scripts> is a comma-separated list of script-files or
           script-categories.
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
  --proxies <url1,[url2],...>: Relay connections through HTTP/SOCKS4 proxies
  --data <hex string>: Append a custom payload to sent packets
  --data-string <string>: Append a custom ASCII string to sent packets
  --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)
  --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 (https://nmap.org/book/man.html) FOR MORE OPTIONS AND EXAMPLES
```

What separates Nmap from other scanners:

The best thing about Nmap is it’s free and open source and is very flexible and versatile. Nmap is often used to determine alive hosts in a network, open ports on those hosts, services running on those open ports, and version identification of that service on that port. It can also run vulnerability assessment scripts to determine if a service is vulnerable. It can be used by penetration testers to identify open ports to gather more information about a target, or can be used by a security administrator to identify open ports in their systems but are not in use. With its usefulness, the security community have long accepted that Nmap is the de-facto tool for network mapping and host identification.

How does Nmap work

Nmap works by sending lots of packets to the target, and then the target responds in a specific way. For example, Nmap may send a packet to the target, specified to port 80(which usually is an HTTP service). Now the target responds specifically, which corresponds to a state(open,closed,filtered). If the port is open, Nmap identifies based on the response what type of HTTP service is running(Apache, Nginx, IIS, etc.). Then another check is used to identify the version, and it goes on.

Nmap has a record of services which maps to a type of response to that type of request, eventually leading to Nmap determining correctly what service and its version. This not foolproof though, as Nmap sometimes mistakes a service running on that port (if you don't do a much more tedious check using the flags) because that port is commonly used for a specific service. **Note** that Nmap scans are better ran with administrator or root privileges as Nmap sends raw packets and being a privileged user is a requirement to send raw packets(as they are far down below the OSI model).

Nmap syntax:

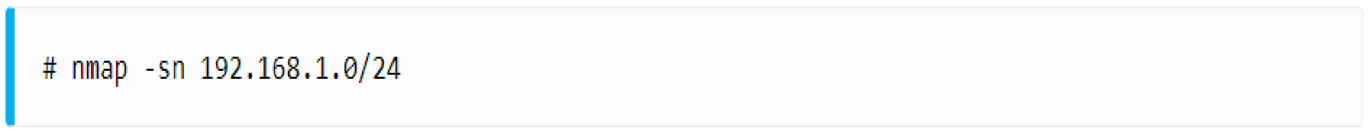
Now before I dive to examples, I want to break down how Nmap syntax is usually structured, as it is good practice to have this by memory. Although in my experience, sequence of flags is not that important(there are cases they are) most of the time. In fact, for a scan to proceed, Nmap only needs a target. This format is vague and is not required, but according to documentation, Nmap follows the format:

```
nmap [scan type] [options] [target specification]
```

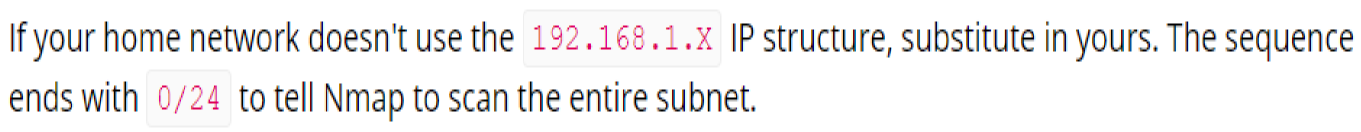
Basic Scans

Nmap has fairly intelligent defaults set, so you are able to just open up Nmap and run a scan without specifying anything but the target. So, why not try it out on a computer on your network. Scanning the computer running Kali isn't going to give you much of anything, so it's best to pick another computer that you own. If you already know the IP of one, awesome. If not, Nmap has a tool to get the IP addresses of the computers on your network.

Open up a terminal, if you haven't already, and run the following linux command.

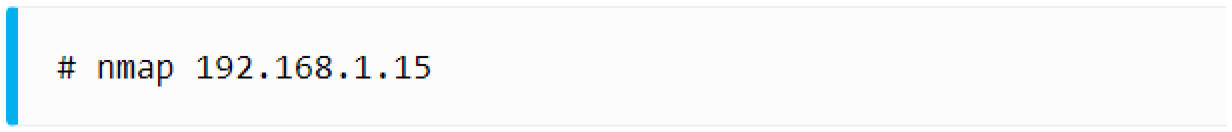


```
# nmap -sn 192.168.1.0/24
```



If your home network doesn't use the `192.168.1.X` IP structure, substitute in yours. The sequence ends with `0/24` to tell Nmap to scan the entire subnet.

What you'll see when Nmap finishes is a list of every devices that was reachable. Each device will have a name(if applicable), IP address, and MAC address with a manufacturer. By using the names and the hardware manufacturers, you should be able to tell what each device on your network is. Pick a computer that you own, and scan it.



```
# nmap 192.168.1.15
```

You can just write in the IP of that computer. Nmap will take a few seconds to probe the computer with packets and report back.

The report will be sort, but it will contain a list of ports with their state and which service they correspond to. It will also show that MAC address information and your IP again.

Useful Flags

Even though the defaults do give some useful information, and you can tell which ports are open, it would still be nice to get some more data. Nmap has tons of flags that you can set to specify just how you would like it to run. There are way too many to cover in this basic guide, but you can always check out Nmap's detailed manpage for more.

-sS

The -sS flag is the default scanning flag for Nmap. It just specifies the way that Nmap will scan. Even though it's the default, it's probably a good idea to specify it anyway.

-T

Timing can be important. Not only does the timing of the scan determine how long scanning will take, but it can also be instrumental in triggering or not triggering firewalls and other safeguards on a target system. While Nmap offers more fine-grained timing control, it also provides a set of six pre-built timing schemes with the -T flag. These timings range from 0 through 5, with 0 being the slowest and least invasive and 5 being the fastest and most overt. -T3 is the default timing flag, but many users prefer -T4 to speed up the scan.

-iL

You can use Nmap to scan multiple targets at once. Doing so can easily be done in-line when you run Nmap.

```
# nmap -sS -T4 192.168.1.4 192.168.1.35 192.168.1.102
```

For a small number of targets this works, but it can quickly become cumbersome and isn't all that readily repeatable. The -iL flag imports a list of targets for Nmap to use. This way, you can save targets and repeat scans at a later date.

Before running Nmap, open up your text editor of choice and enter in a couple of the IPs on your network.

```
$ vim ~/Documents/targets.txt
192.168.1.4
192.168.1.10
192.168.1.35
192.168.1.102
192.168.1.128
```

Save that file and run Nmap with the -iL flag.

```
# nmap -sS -T4 -iL /home/user/Documents/targets.txt
```

Nmap will read through the list and preform a scan on each entry.

-F

By default, Nmap will scan the 1000 most commonly used ports on a target machine. This, of course, takes time. If you know that you only need to or only want to scan the most common ports to reduce the run time of Nmap, you can use the -F flag. The -F flag tells Nmap to only scan the 100 most commonly used ports instead of the usual 1000.

```
# nmap -sS -T4 -F 192.168.1.105
```

-O

If you would like information on the operating system being run on the target machine, you can add the -O flag to tell Nmap to probe for operating system information as well. Nmap is not super accurate when it comes to operating system information, but it usually gets very close.

```
# nmap -sS -T4 -O 192.168.1.105
```

--open

If you are only looking for which ports are open on a specific machine, you can tell Nmap to only look for open ports with the --open flag.

```
# nmap -sS -T4 --open 192.168.1.105
```

-sV

Sometimes, it's useful to know what software and what versions of that software a machine is running. This is especially good for investigating your own servers. It also gives you insight into what server information others can see. Nmap's -sV allows you to get as detailed information as possible about the services running on a machine.

```
# nmap -sS -sV -T4 192.168.1.105
```

-p

Occasionally, you may only want to scan select ports with Nmap. The -p flag allows you to specify specific ports for Nmap to scan. Nmap will then only scan those specified ports on the target machine.

```
# nmap -sS -p- 192.168.1.105
```

This will take a *long* time, so it should not be done lightly.

-A

By now, you've acquired a lot of flags to use. Using all of them together can be very awkward. Nmap has the -A for just this reason. It's sort of the "kitchen sink" flag that tells Nmap to aggressively gather as much information as it can.

```
# nmap -A 192.168.1.105
```

Logging Output

It would sure be able to store the results from Nmap. Well, you can. Nmap has yet another flag that allows you to store output in a variety of different formats. This is excellent for long scans like ones with the -p- flag. To use Nmap's logging capabilities, pass the -oN or -oX along with the name of the file. -oN logs the normal output. -oX logs the output as XML. By default, Nmap will overwrite existing logs with new ones, so be careful not to overwrite anything you don't want to.

```
# nmap -sS -p- -oN  Documents/full-scan.txt 192.168.7.105
```

You can find the full log in the text file when Nmap completes.

If you want something ridiculous, try the -oS flag instead.

Common Nmap scan types:

Nmap have various scan types according to your needs. Common ones are SYN scan (-sS),TCP connect scan(-sT), UDP scan(-sU). Note that the SYN scan and TCP scan utilizes the three way handshake :

1. The **SYN scan** (-sS) identifies a port to be open by sending a “SYN” to the target. If it receives a SYN-ACK or SYN, it marks that port to be open. If it receives a “RST”, it marks the port as filtered.
2. The **TCP scan** (-sT) identifies a port to be open by waiting the completion of the three-way handshake.
3. The **UDP scan** (-sU) is useful for identifying open UDP ports on a target. It sends specific UDP packets to known UDP ports.

Common Nmap options:

Nmap has tons of options depending what you need for your scan. On a basic level, you should be using the following:

1. **Save your Nmap scan to a file.** Nmap can save your scan in 3 formats(normal otput, XML output, and greppable output) using the -oN, -oX, and -oG flags resepctively. If you want to save it in all the formats, use the -oA flag. There exists also a -oS flag which outputs your scan in script kidde or “l33t speak” format.
2. **Add verbosity flags.** Nmap has an option to print out more information using the (-v) flag. You can increase verbosity by printing to standard out (-vv, -vvv or -v2,-v3) its findings during the scan(such as identified open ports, scanning issues, etc.). You can also use the reason flag (double-dash reason). These flags helps you to plan what to do next since you have preliminary data as Nmap doesn’t show its results until after the scan.
3. **Utilize timing options.** You can use a timing template that suits your purpose by using one of the flags -T0,-T1,-T2,-T3,-T4,-T5 which stands for paranoid,sneaky,polite,normal,aggressive, and insane respectively. These templates determine how aggressive your scan will be and is useful depending on the bandwidth and resources of the network you are in. -T0 and -T1 scans are useful for IDS evasion, but may be too slow. T3 or normal is Nmap’s default behavior when none of these flags are mentioned.

Common Nmap target specification:

Nmap allows various target specifications, but can simple be divided into 2 parts: **port** and **IP/host**.

1. Nmap allows **port specification** using the “**-p”** flag. You can specify a single port, port range, excluded ports, and ports for a specific protocol:

```
# Specifying a single port(port 21):
nmap -p21 10.10.10.10
# Specifying a port range(port 21 to 1000):
nmap -p21-1000 10.10.10.10
# Specifying a port range but exclude port 22:
nmap -p21-1000 -exclude-ports 22 10.10.10.10
# Specifying a TCP-specific or UDP-specific port:
# nmap -sU -sS -p T:21,22,25,80,U:53,161 10.10.10.10
```

Note: For you to scan TCP-specific and UDP-specific ports at the same time, it requires that -sU flag is present and at least one scan type for TCP(-sT, -sF, or-sS)

2. Nmap accepts **host names, IP addresses,network range**, **etc.** You can also feed it a file using the -iL flag which contains a list of IP addresses or networks. It also allows you to exclude a specific host or network from your targets:

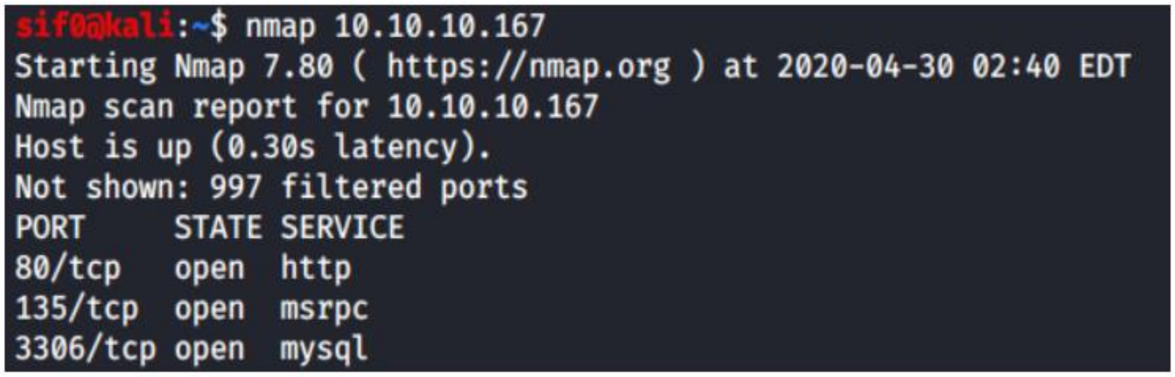
```
# Scanning a specific hostname:
nmap target.domain.com
# Scanning a specific IP address:
nmap 10.10.10.10
# Scanning a network range:
nmap 10.10.10.1-254 or nmap 10.10.10.0/24
# Scanning a network range but exclude few IP addresses:
nmap 10.10.10.0/24 --exclude 10.10.10.167
```

Note: Nmap scans the top 1000 common and mostly used TCP ports if you don't specify a specific port.

Putting it all together:

Here are a few specific examples of Nmap usage:

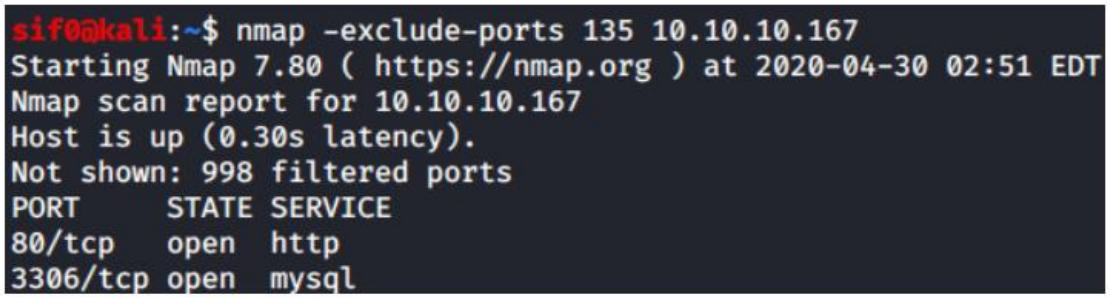
```
Scanning an IP address with no flags used:
```



```
sif0@kali:~$ nmap 10.10.10.167
Starting Nmap 7.80 ( https://nmap.org ) at 2020-04-30 02:40 EDT
Nmap scan report for 10.10.10.167
Host is up (0.30s latency).
Not shown: 997 filtered ports
PORT     STATE SERVICE
80/tcp   open  http
135/tcp  open  msrpc
3306/tcp open  mysql
```

It mentions that out of the top 1000 TCP ports, 997 are filtered and 3 of them are open. Nmap was able to identify that port 80 runs http, 135 runs msrpc, and 3306 runs mysql.

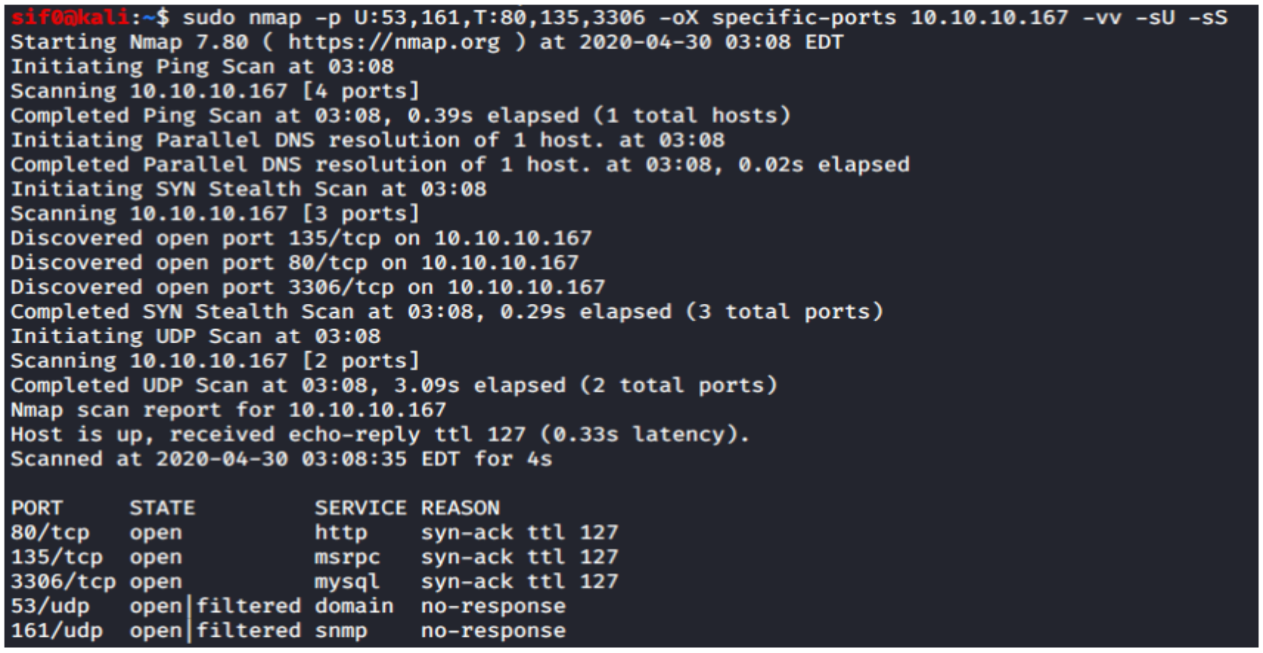
```
Scanning an IP address but excluding a specific TCP port:
```



```
sif0@kali:~$ nmap -exclude-ports 135 10.10.10.167
Starting Nmap 7.80 ( https://nmap.org ) at 2020-04-30 02:51 EDT
Nmap scan report for 10.10.10.167
Host is up (0.30s latency).
Not shown: 998 filtered ports
PORT     STATE SERVICE
80/tcp   open  http
3306/tcp open  mysql
```

Scanning the same host, the scan output only found 2 ports, because I purposefully skipped port 135.

```
Scanning an IP address targeting a specific TCP and UDP port and
saving it to a XML file called "specific-ports" with verbosity of
level 2:
```



```
sif0@kali:~$ sudo nmap -p U:53,161,T:80,135,3306 -oX specific-ports 10.10.10.167 -vv -sU -sS
Starting Nmap 7.80 ( https://nmap.org ) at 2020-04-30 03:08 EDT
Initiating Ping Scan at 03:08
Scanning 10.10.10.167 [4 ports]
Completed Ping Scan at 03:08, 0.39s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 03:08
Completed Parallel DNS resolution of 1 host. at 03:08, 0.02s elapsed
Initiating SYN Stealth Scan at 03:08
Scanning 10.10.10.167 [3 ports]
Discovered open port 135/tcp on 10.10.10.167
Discovered open port 80/tcp on 10.10.10.167
Discovered open port 3306/tcp on 10.10.10.167
Completed SYN Stealth Scan at 03:08, 0.29s elapsed (3 total ports)
Initiating UDP Scan at 03:08
Scanning 10.10.10.167 [2 ports]
Completed UDP Scan at 03:08, 3.09s elapsed (2 total ports)
Nmap scan report for 10.10.10.167
Host is up, received echo-reply ttl 127 (0.33s latency).
Scanned at 2020-04-30 03:08:35 EDT for 4s

PORT     STATE         SERVICE REASON
80/tcp   open          http    syn-ack ttl 127
135/tcp  open          msrpc   syn-ack ttl 127
3306/tcp open          mysql   syn-ack ttl 127
53/udp   open|filtered domain  no-response
161/udp  open|filtered snmp    no-response
```

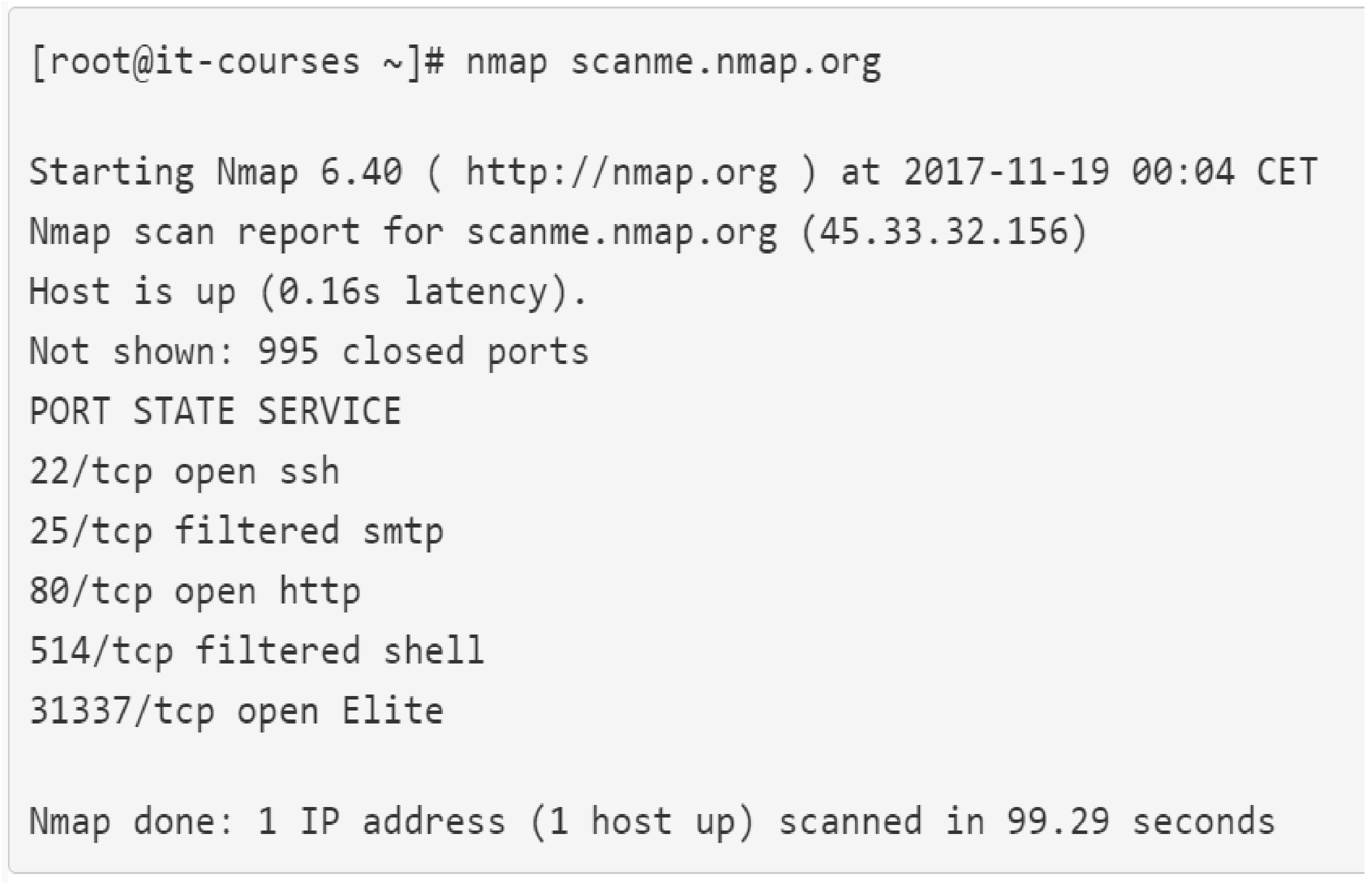
I was able to scan both TCP and UDP ports and got notified that a part is open even while the scan is still running, and saved it to an XML file for future use.

And Then ;

Nmap is a very useful tool not only for security professionals but for IT practitioners because of its versatility.

First scan

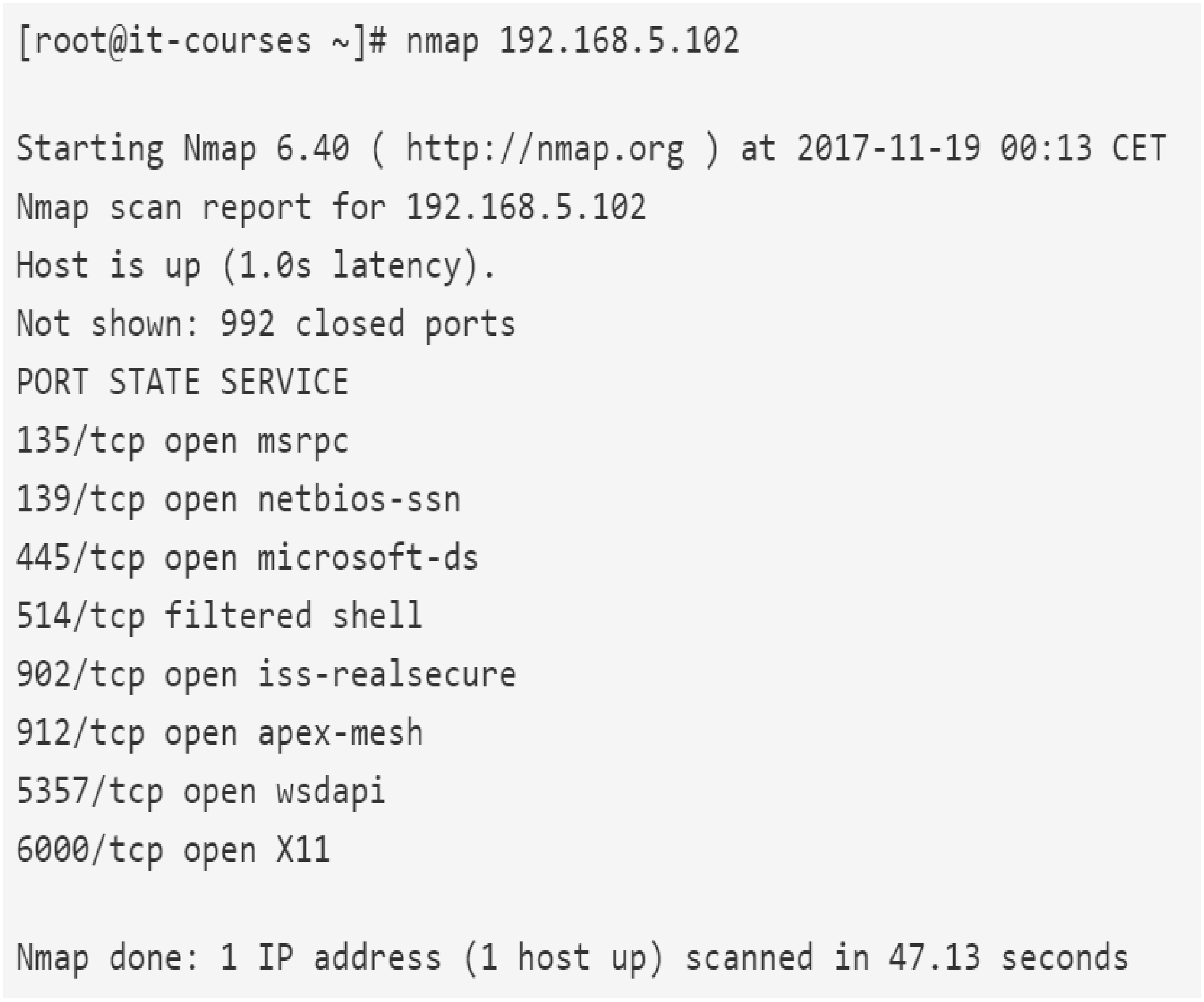
Let’s run our first Nmap scan. The official Nmap website offers a target host at scanme.nmap.org that can be scanned as an example. To scan this machine with all the default parameters, simply run the *nmap scanme.nmap.org* command:



```
[root@it-courses ~]# nmap scanme.nmap.org

Starting Nmap 6.40 ( http://nmap.org ) at 2017-11-19 00:04 CET
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.16s latency).
Not shown: 995 closed ports
PORT STATE SERVICE
22/tcp open ssh
25/tcp filtered smtp
80/tcp open http
514/tcp filtered shell
31337/tcp open Elite

Nmap done: 1 IP address (1 host up) scanned in 99.29 seconds
```

You can probably figure out the output above – three TCP ports are open. The line *22/tcp open ssh* indicates that the TCP port 22 is open, and that the *ssh* service is probably running on that port.

The scan above works only if your system has Internet access. Of course, you can also scan local hosts. Here is an example:



```
[root@it-courses ~]# nmap 192.168.5.102

Starting Nmap 6.40 ( http://nmap.org ) at 2017-11-19 00:13 CET
Nmap scan report for 192.168.5.102
Host is up (1.0s latency).
Not shown: 992 closed ports
PORT STATE SERVICE
135/tcp open msrpc
139/tcp open netbios-ssn
445/tcp open microsoft-ds
514/tcp filtered shell
902/tcp open iss-realsecure
912/tcp open apex-mesh
5357/tcp open wsdapi
6000/tcp open X11

Nmap done: 1 IP address (1 host up) scanned in 47.13 seconds
```

As you can see from the output above, I’ve scanned the local host with an IP address of 192.168.5.102 (it is a Windows 10 host).

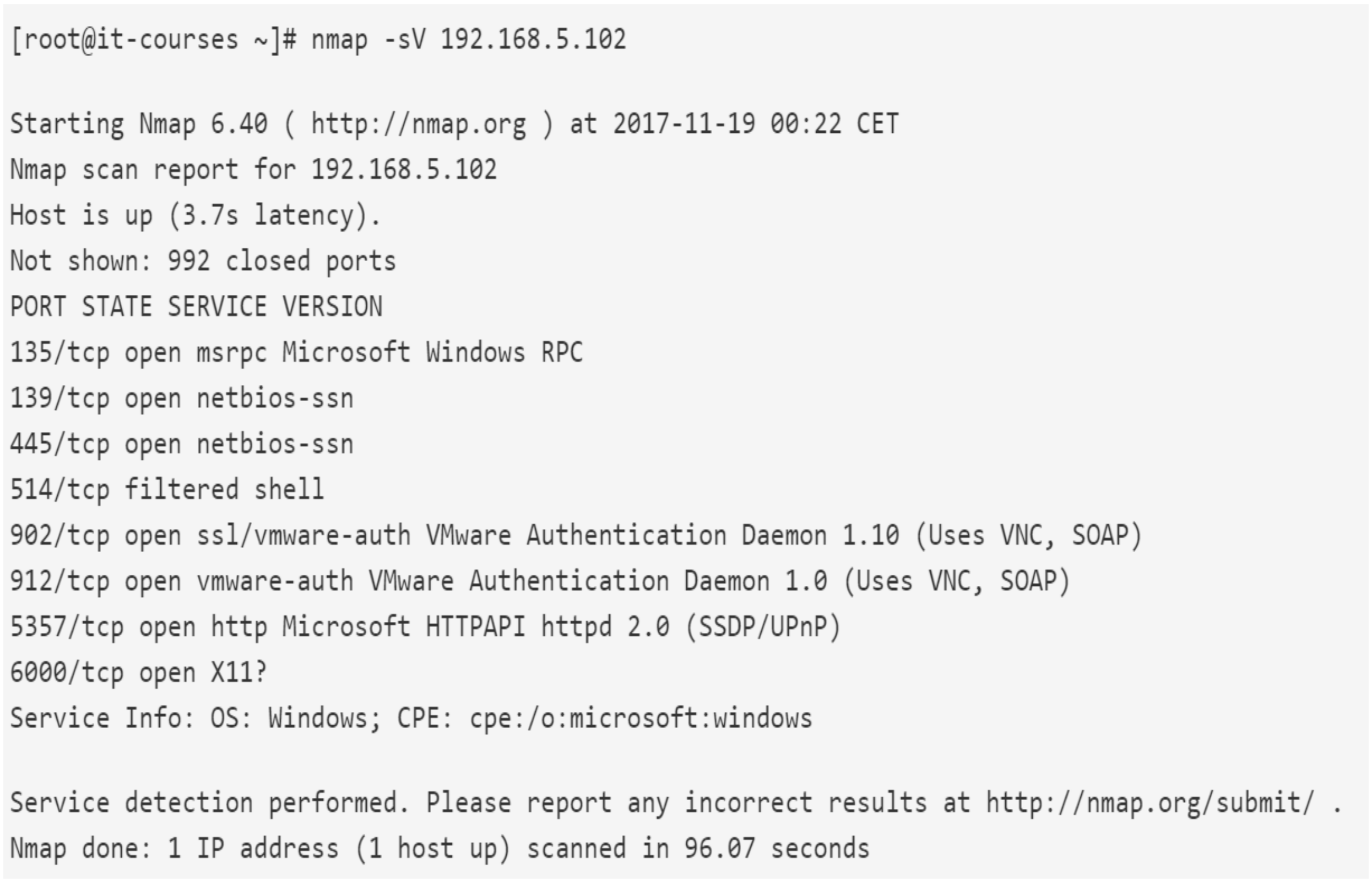
Determine service version

You can also use Nmap to determine the version of the software that the target host is running. This can be particularly useful when doing vulnerability assessments, since you really want to know, for example, which web or DNS servers and versions are running on the host, and having the accurate versions helps dramatically in determining which exploits a server is vulnerable to.

You can determine various interesting information using service scans, including:

- the service protocol (e.g. FTP, SSH, Telnet, HTTP).
- the application name (e.g. BIND, Apache httpd).
- the version number.
- the hostname.
- device type (e.g. printer, router).
- the OS family (e.g. Windows, Linux).

To run a service version scan, we use the *-sV* flags:



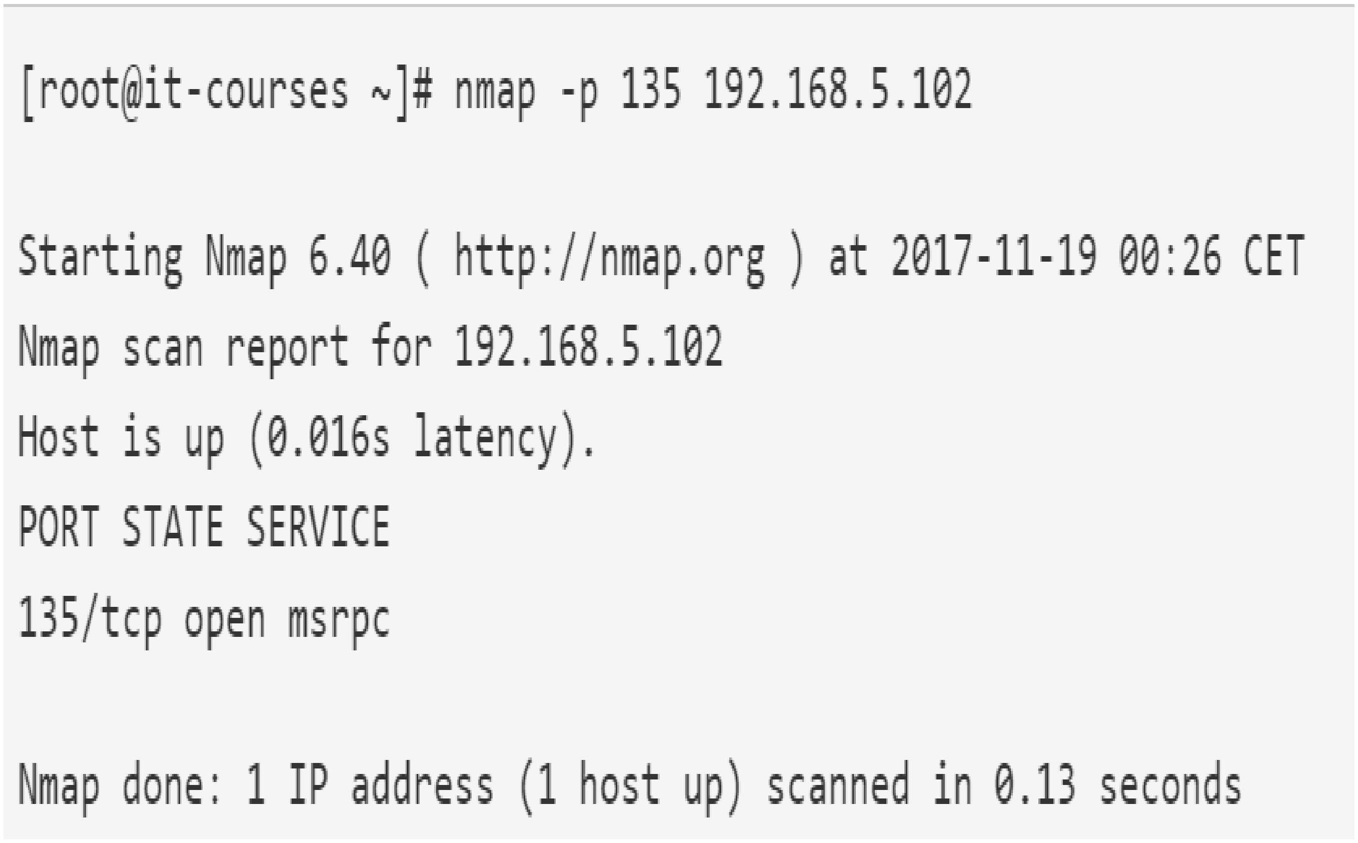
```
[root@it-courses ~]# nmap -sV 192.168.5.102

Starting Nmap 6.40 ( http://nmap.org ) at 2017-11-19 00:22 CET
Nmap scan report for 192.168.5.102
Host is up (3.7s latency).
Not shown: 992 closed ports
PORT STATE SERVICE VERSION
135/tcp open msrpc Microsoft Windows RPC
139/tcp open netbios-ssn
445/tcp open netbios-ssn
514/tcp filtered shell
902/tcp open ssl/vmware-auth VMware Authentication Daemon 1.10 (Uses VNC, SOAP)
912/tcp open vmware-auth VMware Authentication Daemon 1.0 (Uses VNC, SOAP)
5357/tcp open http Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
6000/tcp open X11?
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 96.07 seconds
```

Notice how I got more information about services running on the open ports, including their versions.

Scan specific ports

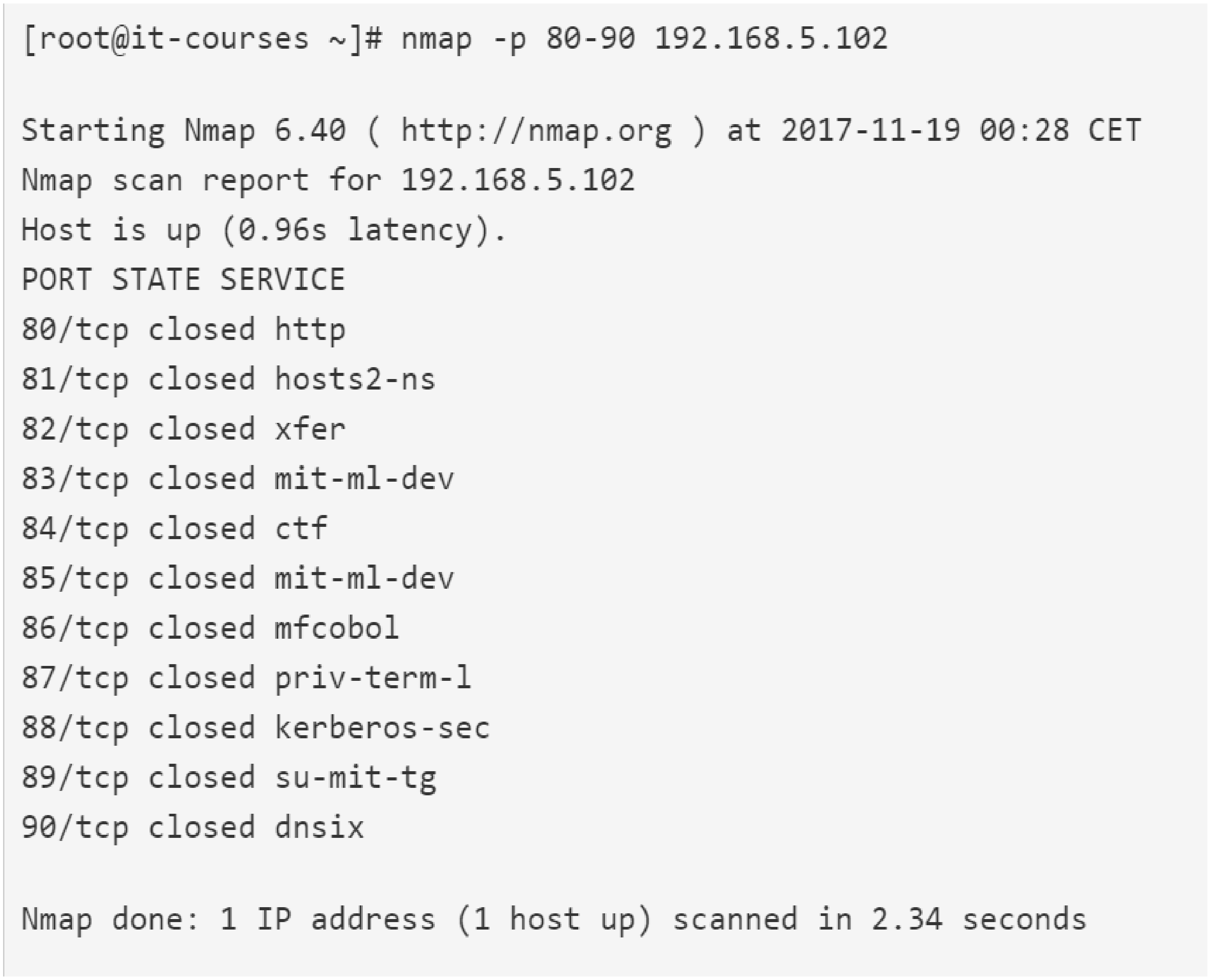
By default, Nmap scans the most common 1,000 ports for each protocol. However, since there are 65535 ports that can be used for service, sometimes you might want to scan very high ports or even individual ports. To do this, the *-p* flag is used.

Here is a simple example. To scan only the port 135, we can use the following command:



```
[root@it-courses ~]# nmap -p 135 192.168.5.102

Starting Nmap 6.40 ( http://nmap.org ) at 2017-11-19 00:26 CET
Nmap scan report for 192.168.5.102
Host is up (0.016s latency).
PORT STATE SERVICE
135/tcp open msrpc

Nmap done: 1 IP address (1 host up) scanned in 0.13 seconds
```

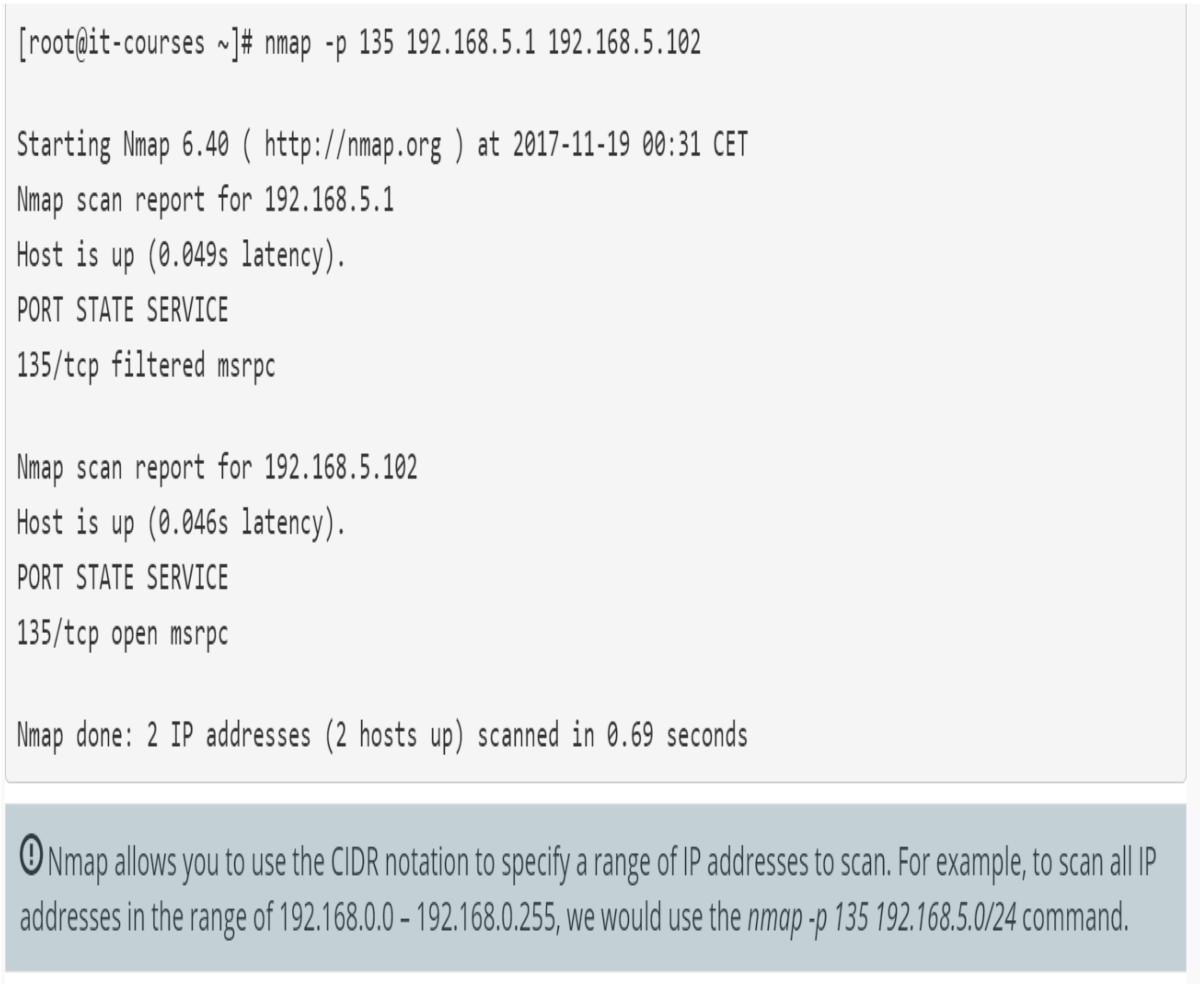
You can also scan a range of ports. This can be done by using the hyphen (-) to specify the range. For example to scan only ports 80 to 90, we would use the following command:



```
[root@it-courses ~]# nmap -p 80-90 192.168.5.102

Starting Nmap 6.40 ( http://nmap.org ) at 2017-11-19 00:28 CET
Nmap scan report for 192.168.5.102
Host is up (0.96s latency).
PORT STATE SERVICE
80/tcp closed http
81/tcp closed hosts2-ns
82/tcp closed xfer
83/tcp closed mit-ml-dev
84/tcp closed ctf
85/tcp closed mit-ml-dev
86/tcp closed mfcobol
87/tcp closed priv-term-l
88/tcp closed kerberos-sec
89/tcp closed su-mit-tg
90/tcp closed dnsix

Nmap done: 1 IP address (1 host up) scanned in 2.34 seconds
```

Specify IP address range

You can scan multiple hosts using a single Nmap command. For example, to scan the port 135 on two hosts with IP addresses of 192.168.5.1 and 192.168.5.102, we would use the following command:



```
[root@it-courses ~]# nmap -p 135 192.168.5.1 192.168.5.102

Starting Nmap 6.40 ( http://nmap.org ) at 2017-11-19 00:31 CET
Nmap scan report for 192.168.5.1
Host is up (0.049s latency).
PORT STATE SERVICE
135/tcp filtered msrpc

Nmap scan report for 192.168.5.102
Host is up (0.046s latency).
PORT STATE SERVICE
135/tcp open msrpc

Nmap done: 2 IP addresses (2 hosts up) scanned in 0.69 seconds
```

Nmap allows you to use the CIDR notation to specify a range of IP addresses to scan. For example, to scan all IP addresses in the range of 192.168.0.0 – 192.168.0.255, we would use the *nmap -p 135 192.168.5.0/24* command.

Finally, that's it. You can play around with different flags and different combinations of flags to tune Nmap to get the exact output that you want. Just be sure to only do this on your own machines and networks, and you will have an interesting experience that just may save your data.

REFERENCES

- scanme.nmap.org
- linuxconfig.org
- medium.com.tr
- Altuntaş Abdulaziz, Kali Linux, Kodlab Yayıları, (2010)