Hands-on Lab Description

CS-CNS-20001 – Network and Security Tool: Nmap

CONTENTS

1 Background of Nmap 4
1.1 What is Nmap? 4
1.2 How to use Nmap? 4
2 Task 1: Nmap Scan Types 5
2.1 TCP Full Connect (Vanilla) Scan 5
2.2 UDP Scan 6
2.3 SYN Scan 6
2.4 FIN Scan 7
2.5 ACK Scan 7
2.6 NULL Scan 8
2.7 XMAS Scan 8
2.8 RPC Scan 9
2.9 IDLE Scan 9
2.10 Summary of Nmap Scan Techniques 10
3 Task 2 Running Nmap Scripts 12
3.1 Run Nmap's Most Popular Scripts 13
3.2 Run all the scripts within a category 15
3.3 Run Nmap scripts with a wildcard * 16
3.4 Run a single Nmap script 16
3.5 Run your own scripts 16
4 Related Information and Resource 16

Category:

```
CS-CNS: Computer Network Security
```

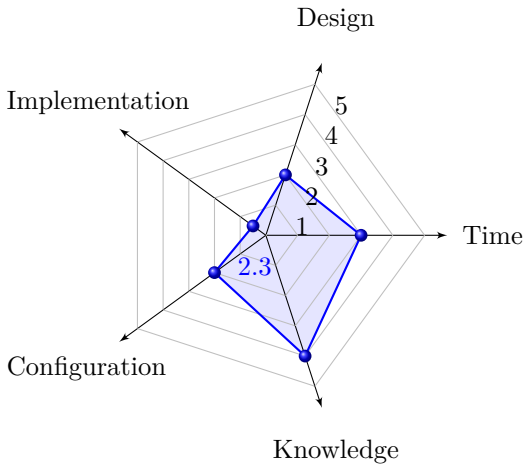
Objectives:

```
Understand network testing tool hping3
Use hping3 for network scanning and emulating DoS attacks
```

Estimated Lab Duration:

```
Expert: 120 minutes
Novice: 480 minutes
```

Difficulty Diagram:



Difficulty Table.

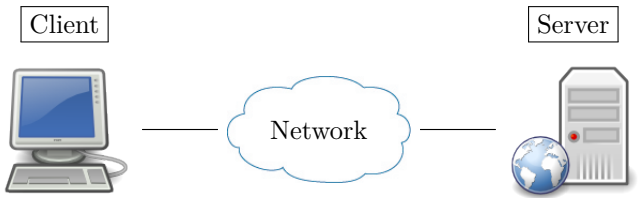
Measurements	Values (0-5)
Time	3
Design	2
Implementation	0.5
Configuration	2
Knowledge	4
Score (Average)	2.3

Required OS:

```
Linux: Kali 20.3
```

Lab Running Environment:

```
ThoTh Lab: https://thothlab.org
```



```
Client: Linux (Kali 20.3)
Server: Linux (Metasploitable-2)
Network Setup:
Internet is connected through Network: 192.168.0.0/24
```

Lab Preparations:

```
Know how to use Linux OS (Reference Labs: CS-SYS-00001)
Basic knowledge about computer networking (Reference Labs: CS-CNS-10003 Hping3)
```

Lab Overview

Nmap is an open-source tool used for network discovery and security auditing. In short, Nmap displays exposed services on a target machine along with other useful information such as the version and OS detection.

In summary, students will do:

- Use nmap to send customized packets for network diagnostic and testing
- Use nmap to send customized packets for emulating attacks

1 Background of Nmap

Nmap is a network mapper that has emerged as one of the most popular, free network discovery tools on the market. Nmap is now one of the core tools used by network administrators to map their networks. The program can find live hosts on a network, perform port scanning, ping sweeps, OS detection, and version detection.

1.1 What is Nmap?

Nmap is a network scanning tool that uses IP packets to identify all the devices connected to a network and provide information on the services and operating systems they are running.

The program is most commonly used via a command-line interface (though GUI front-ends are also available). It is available for many different operating systems such as Linux, Free BSD, and Gentoo. Its popularity has also been bolstered by an active and enthusiastic user support community. Nmap was developed for enterprise-scale networks and can scan through thousands of connected devices. Moreover, Nmap has been increasingly used to inspect IoT security. Generally speaking, Nmap provides information on:

1. Every active IP so you can determine if an IP is being used by a legitimate service or an external attacker.
2. You network as a whole, including live hosts, open ports, and the OS of every connected device.
3. Vulnerabilities, scan your own server to simulate the process that a hacker would use to attack your site.

The primary uses of Nmap can be broken into three core processes. First, the program gives you detailed information on every IP active on your networks, and each IP can then be scanned. This allows administrators to check whether an IP is being used by a legitimate service or an external attacker.

Secondly, Nmap provides information on your network as a whole. It can be used to provide a list of live hosts and open ports and identify the OS of every connected device. This makes it a valuable tool in ongoing system monitoring and a critical part of pentesting. Nmap can be used alongside the Metasploit framework, for instance, to probe and then repair network vulnerabilities.

Thirdly, Nmap has also become a valuable tool for users looking to protect personal and business websites. Using Nmap to scan your own web server, particularly if you are hosting your website from home, is essentially simulating a hacker's process to attack your site. 11Attacking" your own site in this way is a powerful way of identifying security vulnerabilities.

1.2 How to use Nmap?

To check if Nmap is installed on your system, you can check the following command:

```
$ nmap -v
```

If Nmap is installed, it may generate outputs like:

```
Starting Nmap 7.91 ( https://nmap.org ) at 2020-10-26 11:41 MST
Read data files from: /usr/bin/../share/nmap
Nmap done: 0 IP addresses (0 hosts up) scanned in 0.05 seconds
```

If Nmap is not installed, you can issue the following command on your Linux CLI prompt:

```
$ sudo apt install nmap
```

Suggestions:

1. Review and exercise the following labs CS-SYS-00001 (Linux tutorial), CS-NET-00001 (Network setup) and CS-NET-00002(Gateway setup).

Note that refer to related lab CNS-100003 (hping3). Many concepts of scanning functions of hping3 can be applied in this lab.

Nmap is straightforward to use, and most of the tools it provides are familiar to system admins from other programs. Nmap's advantage is that it brings a wide range of these tools into one program, rather than forcing you to skip between separate and discrete network monitoring tools.

To use Nmap, you need to be familiar with command-line interfaces. Most advanced users can write scripts to automate common tasks, but this is unnecessary for basic network monitoring.

2 Task 1: Nmap Scan Types

In the following examples, we assume that the scanning target's IP address is 192.168.0.4.

2.1 TCP Full Connect (Vanilla) Scan

TCP Connect scan completes the 3-way handshake. If a port is open, the operating system completed the TCP three-way handshake, and the port scanner immediately closes the connection to avoid DOS. This is "noisy" because the services can log the sender IP address and trigger Intrusion Detection Systems.

Syntax of Full Scan:

```
$ nmap -sT target_IP
```

Nmap Full scan example:

```
$ nmap -sT 192.168.0.4
```

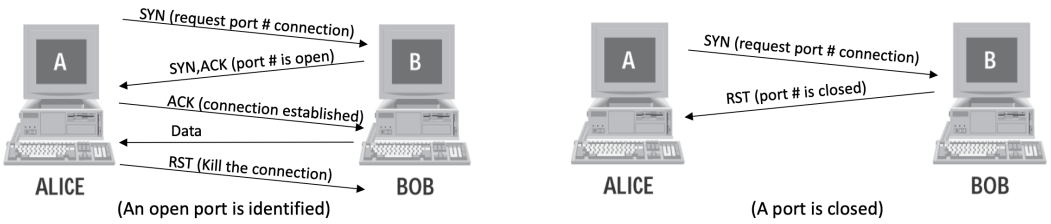


Figure CS-CNS-20001.1
Full Scan.

As shown in Figure CS-CNS-20001.1, the Full scan has the following features:

- Attempts to completes 3-way handshake with each scanned port.
- Sends SYN and waits for ACK before sending ACK.
- Tears down connection using FIN packets.
- If target port is closed, sender will received either no response, a RESET packet , or an ICMP Port Unreachable packet.
- Not stealthy, the connection request can be logged.

2.2 UDP Scan

This scan checks to see if any UDP ports are listening. Since UDP does not respond with a positive acknowledgment like TCP and only responds to an incoming UDP packet when the port is closed, this type of scan can sometimes show false positives. However, it can also reveal Trojan horses running on high UDP ports and hidden RPC services. It may be quite slow since some machines intentionally slow down responses to this kind of traffic to avoid being overwhelmed. Machines running Windows OS, however, do not implement this slowdown feature, so you should use UDP to scan Windows hosts normally.

Syntax of UDP Scan:

```
$ nmap -sU target_IP
```

Nmap UDP scan example:

```
$ nmap -sU 192.168.0.4
```

The UDP scan has the following features:

- Sends a UDP packet to target ports to determine if a UDP service is listening.
- If the target system returns an ICMP Port Unreachable message, the target port is closed. Otherwise, the target port is assumed to be open.
- Unreliable since there may be false positives.
- Client program of a discovered open port is used to verify service.

The UDP scan can be very slow due to the following reasons:

- Open and filtered ports rarely send any response, leaving Nmap to time out.
- Closed ports usually send back an ICMP port unreachable error. But many hosts rate limit ICMP port unreachable messages by default (e.g., 1/sec).

To speed up the UDP scan, you can apply the following approaches:

- Increase host parallelism: use –min-hostgroup to scan multiple hosts at once.
- Scan popular ports first: using the -F option to scan the most common UDP ports.
- Add –version-intensity 0 to version detection scans.
- Scan from behind the firewall.
- Use –host-timeout to skip slow hosts.
- Use -v (verbosity) and chill out, Nmap provides estimated time for scan completion of each host.

2.3 SYN Scan

SYN scan is another form of TCP scanning. Rather than using the operating system's network functions, the port scanner generates raw IP packets and monitors for responses. This scan type is also known as "half-open scanning" because it never actually opens a full TCP connection. The port scanner generates an SYN packet. If the target port is open, it will respond with an SYN-ACK packet. The scanner host responds with an RST packet, closing the connection before the handshake is completed. If the port is closed but unfiltered, the target will instantly respond with an RST packet. There is debate over which scan is less intrusive on the target host. SYN scan has the advantage that the individual services never actually receive a connection.

Syntax of SYN Scan:

```
$ nmap -sS target_IP
```

Nmap SYN scan example:

```
$ nmap -sS 192.168.0.4
```

Figure CS-CNS-20001.2
SYN Scan.

As shown in Figure CS-CNS-20001.2, the SYN scan has the following features:

- Only sends the initial SYN and waits for ACK to detect open port.
- SYN scans stop two-thirds of the way through the 3-way handshake, a.k.a., half-open scan.
- Attacker sends a RESET (RST) after receiving an SYN-ACK response.
- A true connection is never established; thus, the target may not log the connection.
- Faster and stealthier than a full TCP Connect scans.
- If the target port is closed, the destination will send a RESET or nothing.
- SYN flood may cause accidental denial-of-service attacks if the target is slow.

2.4 FIN Scan

This is a stealthy scan, like the SYN scan, but sends a TCP FIN packet instead. Most, but not all computers will send an RST packet back if they get this input so that the FIN scan can show false positives and negatives, but it may get under the radar of some IDS programs and other countermeasures.

Syntax of FIN Scan:

```
$ nmap -sF target_IP
```

Nmap FIN scan example:

```
$ nmap -sF 192.168.0.4
```

2.5 ACK Scan

Ack scanning determines whether the port is filtered or not. This is especially good when attempting to probe for the existence of a firewall and its rulesets. Simple packet filtering will allow established connections (packets with the ACK bit set), whereas a more sophisticated stateful firewall might not.

Syntax of SYN Scan:

```
$ nmap -sA target_IP
```

Nmap ACK scan example:

```
$ nmap -sA 192.168.0.4
```

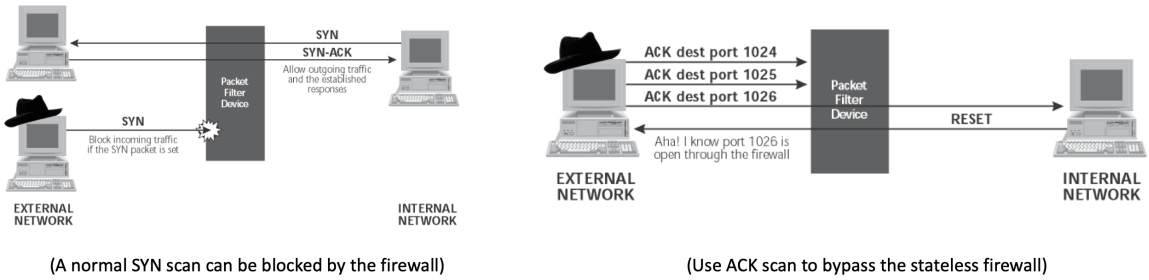


Figure CS-CNS-20001.3
Full Scan.

As shown in Figure CS-CNS-20001.3, the ACK scan has the following features:

- Sends a packet with the ACK code bit set to each target port.
- Allows the attacker to get past some packet filtering devices.
- Allows the attacker to determine what kind of established connections a firewall or router will allow into a network by determining which ports through a firewall allow established connection responses.
- If no response or an ICMP Port Unreachable message is returned, Nmap will label the target port as "filtered," meaning that a packet filter is blocking the response.

2.6 NULL Scan

Another very stealthy scan that sets all the TCP header flags to off or null. This is not normally a valid packet, and some hosts will not know what to do with this. Windows operating systems are in this group, and scanning them with NULL scans will produce unreliable results. However, for non-Windows servers protected by a firewall, this can be a way to get through.

Syntax of Null Scan:

```
$ nmap -sN target_IP
```

Nmap Null scan example:

```
$ nmap -sN 192.168.0.4
```

2.7 XMAS Scan

Similar to the NULL scan, except for all the flags in the TCP header is set to on. Windows machines won't respond to this due to the way their TCP stack is implemented. Xmas scans derive their name from the set

of flags that are turned on within a packet. These scans are designed to manipulate the PSH, URG, and FIN flags of the TCP header.

Syntax of Xmas Scan:

```
$ nmap -sX target_IP
```

Nmap Xmas scan example:

```
$ nmap -sX 192.168.0.4
```

2.8 RPC Scan

This special type of scan looks for machines answering to RPC (Remote Procedure Call) services. RPC, which allows remote commands to be run on the machine under certain conditions, can be a dangerous service. Since RPC services can run on many different ports, it is hard to tell from a normal scan which ones might be running RPC. This scan will probe the ports found open on a machine with commands to show the program name and version if RPC is running. It's not a bad idea to run one of these scans every often to find out if and where you have these services running.

Syntax of RPC Scan:

```
$ nmap -sR target_IP
```

Nmap RPC scan example:

```
$ nmap -sR 192.168.0.4
```

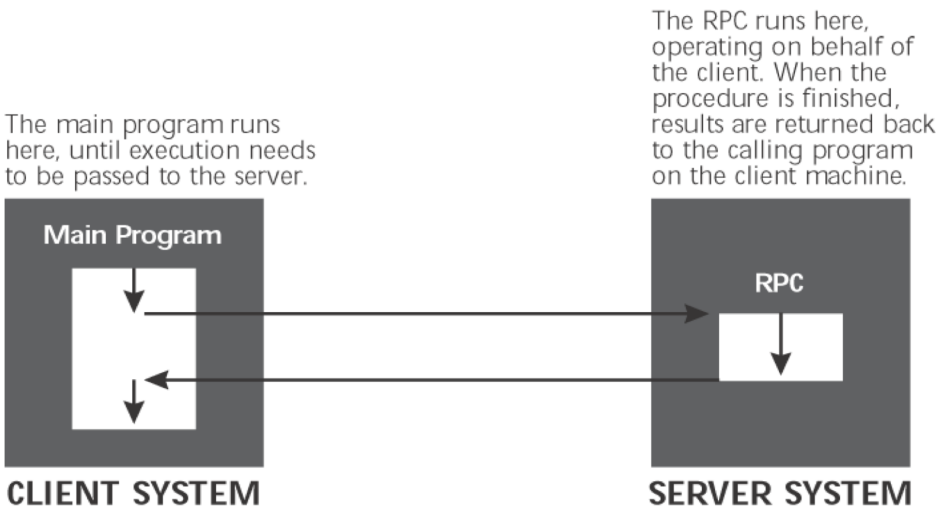


Figure CS-CNS-20001.4
Full Scan.

As shown in Figure CS-CNS-20001.4, the RPC scan has the following features:

- Scans RPC services using all discovered open TCP/UDP ports on the target to send RPC NULL commands. Tries to determine if an RPC program is listening at the port and identifies type of RPC program .

2.9 IDLE Scan

It is a super stealthy method whereby the scan packets are bounced off an external host. You don't need to have control over the other host, but it does have to set up and meet certain requirements. You must input the IP address of our "zombie" host and what port number to use. It is one of the more controversial options in Nmap since it really only has a use for malicious attacks.

Syntax of RPC Scan:

```
$ nmap -sL zombin_IP:port target_IP
```

Nmap Idle scan example:

```
$ nmap -sI 192.168.0.1 192.168.0.4
```

In this example, 192.168.0.1 is the zombie host. The procedure of Idle scan is summarized as follows:

1. Probe the zombie's IP ID and record it. Forge an SYN packet from the zombie and send it to the desired port on the target.
2. Depending on the port state, the target's reaction may or may not cause the zombie's IP ID to be incremented.
3. Probe the zombie's IP ID again. The target port state is then determined by comparing this new IP ID with the one recorded in step 1.

2.10 Summary of Nmap Scan Techniques

In follows, we assume that the scanning target's IP address is 192.168.0.4.

```
+--------+----------------------+---------------------+
| Switch |     Description      |       Example       |
+--------+----------------------+---------------------+
|   -sS  |   TCP SYN port scan. | nmap -sS 192.168.0.4 |
+--------+----------------------+---------------------+
|   -sT  | TCP Connect port scan. | nmap -sT 192.168.0.4 |
+--------+----------------------+---------------------+
|   -sU  |    UDP port scan.    | nmap -sU 192.168.0.4 |
+--------+----------------------+---------------------+
|   -sA  |   TCP ACK port scan. | nmap -sA 192.168.0.4 |
+--------+----------------------+---------------------+
|   -sN  |   TCP Null port scan. | nmap -sN 192.168.0.4 |
+--------+----------------------+---------------------+
|   -sF  |   TCP FIN port scan. | nmap -sF 192.168.0.4 |
+--------+----------------------+---------------------+
|   -sX  |   TCP Xmas port scan. | nmap -sX 192.168.0.4 |
+--------+----------------------+---------------------+
|   -sI  |   TCP Idle port scan. | nmap -sI IP_1 IP_2 |
+--------+----------------------+---------------------+
|   -sR  |   TCP RPC port scan. | nmap -sR 192.168.0.4 |
+--------+----------------------+---------------------+
|   -sP  |   TCP Ping scan.     | nmap -sP 192.168.0.4 |
+--------+----------------------+---------------------+
|   -sO  |   TCP Protocol scan. | nmap -sO 192.168.0.4 |
+--------+----------------------+---------------------+
|   -b   |   FTP Bounce scan.   | nmap -b ftp_IP target|
+--------+----------------------+---------------------+
```

Host Discovery

Switch	Description	Example
-Pn	Only port scan.	nmap -Pn 192.168.0.4
-sn	Only host discovery.	nmap -sn 192.168.0.4
-PR	ARP discovery on local network.	nmap -PR 192.168.0.4
-n	Disable DNS resolution.	nmap -n 192.168.0.4

Port Specification

Switch	Description	Example
-p	Port or port range.	nmap -p 22-80 192.168.0.4
-p-	Scan all ports.	nmap -p- 192.168.0.4
-F	Fast port scan. (top 100)	nmap -F 192.168.0.4

Service and Version Detection

Switch	Description	Example
-sV	Detect the version of services.	nmap -sV 192.168.0.4
-A	Enable OS detection, version detection, script scanning and traceroute.	nmap -A 192.168.0.4

OS Detection

Switch	Description	Example
-O	Identify OS using TCP/IP strack fingerprinting.	nmap -O 192.168.0.4

Timing and Performance

Switch	Description	Example
-T0	Paranoid IDS evasion.	nmap -T0 192.168.0.4
-T1	Sneaky IDS evasion.	nmap -T1 192.168.0.4

```
|   -T2  |       Polite IDS evasion.     | nmap -T2 192.168.0.4 |
|        |    (requires less bandwidth) |                      |
+--------+-------------------------------+----------------------+
|   -T3  |   Normal IDS evasion. (default) | nmap -T3 192.168.0.4 |
+--------+-------------------------------+----------------------+
|   -T4  |     Aggressive speed scan.    | nmap -T4 192.168.0.4 |
|        |     (requires fast network)   |                      |
+--------+-------------------------------+----------------------+
|   -T5  |       Insane speed scan.      | nmap -T5 192.168.0.4 |
|        | (requires massive network speed) |                   |
+--------+-------------------------------+----------------------+
```

NSE Scripts

```
+-----------------+-----------------------+----------------------+
|     Switch      |      Description      |       Example        |
+-----------------+-----------------------+----------------------+
|       -sC       |  Default script scan. | nmap -sC 192.168.0.4 |
+-----------------+-----------------------+----------------------+
| --script banner | Specify single script. | nmap --script banner |
|                 |    (banner grabbing) |     192.168.0.4    |
+-----------------+-----------------------+----------------------+
```

Firewall / IDS Evasion

```
+--------+-------------------------------+----------------------+
| Switch |        Description            |        Example       |
+--------+-------------------------------+----------------------+
|   -f   |   Use fragmented IP packets. | nmap -f 192.168.0.4 |
|        |    (packet filter evasion)  |                      |
+--------+-------------------------------+----------------------+
|   -D   |         Decoy scan.           |   nmap -D 192.168.0.4 |
|        |     (spoofed source IPs)    |                      |
+--------+-------------------------------+----------------------+
|   -g   | Use given source port number. | nmap -g 22 192.168.0.4 |
+--------+-------------------------------+----------------------+
```

3 Task 2 Running Nmap Scripts

NSE stands for Nmap Scripting Engine, and it is basically a digital library of Nmap scripts that help enhance the default Nmap features and report the results in a traditional Nmap output. You can also write and share your own scripts, so you're not limited to relying on the Nmap default NSE scripts. The only requirement for you to write these scripts is that they must be coded using the *Lua programming language*.

Four different types can help us enhance the default Nmap features, depending on the target and the scanning phase in which they are run.

1. Prerule scripts: These types of scripts run before the rest of any scanning operation, while Nmap doesn't have any data about the remote target.

2. Host scripts: Once the Nmap default scan has finished the host exploration, detection, port scanning, or software discovery, it will perform the host scripts.

3. Service scripts: These are a particular set of Nmap scripts that are run against services on the remote host. These include HTTP service scripts, for example, which can be run against web servers.
4. Postrule scripts: These are run after the entire Nmap scan has finished and are often useful for parsing, formatting, and presenting the different results.

Table CS-CNS-20001.1 shows the Nmap script categories.

Table CS-CNS-20001.1
Nmap script categorites.

Nmap Script	Name Description
auth	All sorts of authentication and user privilege scripts
broadcast	Network discovery scripts that use broadcast petitions for intel gathering
brute	Set of scripts for performing brute force attacks to guess access credentials
default	The most popular Nmap scripts, using -sC by default
discovery	Scripts related to network, service and host discovery
dos	Denial of service attack scripts used to test and perform DOS and floods
exploit	Used to perform service exploitation on different CVEs
external	Scripts that rely on 3rd party services or data
fuzzer	Used to perform fussing attacks against apps, services or networks
intrusive	All the 'aggressive' scripts that cause a lot of network noise
malware	Malware detections and exploration scripts
safe	Safe and non-intrusive/noisy scripts
version	OS, service and software detection scripts
vuln	The Nmap vuln category includes vulnerability detection and exploitation scripts

3.1 Run Nmap's Most Popular Scripts

When dealing with over 600 default Nmap scripts, it is not easy to find the most popular ones by inspecting them one by one. That is why the Nmap team has built an '-sC' option, which lets you run the top Nmap scripts at once. To do this, use the '-sC' argument, as shown below:

```
$ nmap -sC 192.168.0.4
```

The following is the output of scanning the Metasploitable 2 host.

```
Starting Nmap 7.91 ( https://nmap.org ) at 2020-10-26 15:18 MST
Nmap scan report for 192.168.0.4
Host is up (0.00065s latency).
Not shown: 977 closed ports
PORT    STATE SERVICE
21/tcp  open  ftp
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 192.168.0.15
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
```

```
|     vsFTPd 2.3.4 - secure, fast, stable
|_End of status
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)
23/tcp  open  telnet
25/tcp  open  smtp
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN,
   STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN,
|_ssl-date: 2020-10-26T22:19:38+00:00; -17s from scanner time.
| sslv2:
|   SSLv2 supported
|   ciphers:
|     SSL2_DES_192_EDE3_CBC_WITH_MD5
|     SSL2_RC4_128_WITH_MD5
|     SSL2_RC4_128_EXPORT40_WITH_MD5
|     SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
|     SSL2_DES_64_CBC_WITH_MD5
|_    SSL2_RC2_128_CBC_WITH_MD5
53/tcp  open  domain
| dns-nsid:
|_  bind.version: 9.4.2
80/tcp  open  http
|_http-title: Metasploitable2 - Linux
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    49188/udp  mountd
|   100005 1,2,3    51176/tcp  mountd
|   100021 1,3,4    38618/tcp  nlockmgr
|   100021 1,3,4    41912/udp  nlockmgr
|   100024 1        48185/tcp  status
|_  100024 1        54825/udp  status
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
| mysql-info:
|   Protocol: 10
|   Version: 5.0.51a-3ubuntu5
|   Thread ID: 21
|   Capabilities flags: 43564
```

```
|   Some Capabilities: SupportsCompression, SupportsTransactions, Support41Auth,
    SwitchToSSLAfterHandshake, Speaks41ProtocolNew, LongColumnFlag,
    ConnectWithDatabase
|   Status: Autocommit
|_  Salt: `hlc/^jyAy^I5k=.n`wV
5432/tcp open postgresql
|_ssl-date: 2020-10-26T22:18:41+00:00; -18s from scanner time.
5900/tcp open vnc
| vnc-info:
|   Protocol version: 3.3
|   Security types:
|_    VNC Authentication (2)
6000/tcp open X11
6667/tcp open irc
8009/tcp open ajp13
|_ajp-methods: Failed to get a valid response for the OPTION request
8180/tcp open unknown
|_http-favicon: Apache Tomcat
|_http-title: Apache Tomcat/5.5

Host script results:
|_clock-skew: mean: 59m42s, deviation: 2h00m00s, median: -17s
|_nbstat: NetBIOS name: METASPLOITABLE-, NetBIOS user: <unknown>, NetBIOS MAC:
    <unknown> (unknown)
| smb-os-discovery:
|   OS: Unix (Samba 3.0.20-Debian)
|   Computer name: Metasploitable-Client
|   NetBIOS computer name:
|   Domain name:
|   FQDN: Metasploitable-Client
|_  System time: 2020-10-26T18:18:27-04:00
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
|_smb2-time: Protocol negotiation failed (SMB2)

Nmap done: 1 IP address (1 host up) scanned in 72.21 seconds
```

3.2 Run all the scripts within a category

If you want to run all the scripts within a category, then you can use the '–script category' name, as you can see here:

```
$ nmap --script discovery 192.168.0.4
```

You can even combine two categories if needed:

```
$ nmap --script default,safe 192.168.0.4
```

Or if you want to exclude some category, you can prevent certain script categories from running—use the 'not' parameter before the category name. as in this example:

```
$ nmap --script "not vuln" 192.168.0.4
```

For nmap 'vuln' scanning results on a metasploitable-2 machine, you can download the scanning results at:

```
https://gitlab.thothlab.org/thoth-group/ThoThLabResource/raw/master/lab-cs-cns-20001.zip
```

3.3 Run Nmap scripts with a wildcard *

Nmap also allows you to run scripts using wildcards, meaning you can target multiple scripts that finish or end up with any pattern. For example, if you want to run all the scripts that begin with 'FTP', you could use this syntax:

```
$ nmap --script "ftp-\*" 192.168.0.4
```

The same goes for SSH:

```
$ nmap --script "ssh-\*" 192.168.0.4
```

3.4 Run a single Nmap script

This is the perfect solution when you already know which script is going to be used. For example, if we want to run the HTTP-brute script to perform brute force password auditing against HTTP basic, digest, and ntlm authentication, we'll use:

```
$ nmap --script="http-brute" 192.168.0.4
```

3.5 Run your own scripts

As we said before, NSE has the ability to let you write your own scripts and run those scripts locally in your operating system. For this purpose, you can use this syntax:

```
$ nmap --script =/your-scripts 192.168.0.4
$ \end{Bash
$
$ Just make sure you replace the path /your-scripts with the local path where your
    scripts are stored.
$
$ \subsection{Combine them all}
$ You've seen how easy it is to run single scripts, categories, and even your own
    local scripts. Now let's combine them all into one single command, as shown in
    the following example:
$ \begin{Bash}
$ nmap --script "vuln,safe,/my/script and not ftp-\*" 192.168.0.4
```

In the previous script, we combined two script categories; your own script and omitting all scripts that begin with "ftp-".

4 Related Information and Resource

```
nmap references and documents: https://nmap.org
TCP Idle scan: https://nmap.org/book/idlescan.html
FTP Bounce scan: https://nmap.org/book/scan-methods-ftp-bounce-scan.html
29 Practical Examples of Nmap: https://www.tecmint.com/nmap-command-examples/
Nmap default scripts: https://nmap.org/nsedoc/
Nmap script writing tutorial: https://nmap.org/book/nse-tutorial.html
```

```
Nmap cheating sheet:
Basic Scanning Techniques
   Scan a single target -> nmap [target]
   Scan multiple targets -> nmap [target1,target2,etc]
   Scan a list of targets -> nmap -iL [list.txt]
   Scan a range of hosts -> nmap [range of IP addresses]
   Scan an entire subnet -> nmap [IP address/cdir]
   Scan random hosts -> nmap -iR [number]
   Excluding targets from a scan -> nmap [targets] -exclude [targets]
   Excluding targets using a list -> nmap [targets] -excludefile [list.txt]
   Perform an aggressive scan -> nmap -A [target]
   Scan an IPv6 target -> nmap -6 [target]

Discovery Options
   Perform a ping scan only -> nmap -sP [target]
   Don't ping -> nmap -PN [target]
   TCP SYN Ping -> nmap -PS [target]
   TCP ACK ping -> nmap -PA [target]
   UDP ping -> nmap -PU [target]
   SCTP Init Ping -> nmap -PY [target]
   ICMP echo ping -> nmap -PE [target]
   ICMP Timestamp ping -> nmap -PP [target]
   ICMP address mask ping -> nmap -PM [target]
   IP protocol ping -> nmap -PO [target]
   ARP ping -> nmap -PR [target]
   Traceroute -> nmap -traceroute [target]
   Force reverse DNS resolution -> nmap -R [target]
   Disable reverse DNS resolution -> nmap -n [target]
   Alternative DNS lookup -> nmap -system-dns [target]
   Manually specify DNS servers -> nmap -dns-servers [servers] [target]
   Create a host list -> nmap -sL [targets]

Advanced Scanning Options
   TCP SYN Scan -> nmap -sS [target]
   TCP connect scan -> nmap -sT [target]
   UDP scan -> nmap -sU [target]
   TCP Null scan -> nmap -sN [target]
   TCP Fin scan -> nmap -sF [target]
   Xmas scan -> nmap -sX [target]
   TCP ACK scan -> nmap -sA [target]
   Custom TCP scan -> nmap -scanflags [flags] [target]
   IP protocol scan -> nmap -sO [target]
   Send Raw Ethernet packets -> nmap -send-eth [target]
   Send IP packets -> nmap -send-ip [target]

Port Scanning Options
```

```
   Perform a fast scan -> nmap -F [target]
   Scan specific ports -> nmap -p [ports] [target]
   Scan ports by name -> nmap -p [port name] [target]
   Scan ports by protocol -> nmap -sU -sT -p U:[ports],T:[ports] [target]
   Scan all ports -> nmap -p "*" [target]
   Scan top ports -> nmap -top-ports [number] [target]
   Perform a sequential port scan -> nmap -r [target]

Version Detection
   Operating system detection -> nmap -O [target]
   Submit TCP/IP Fingerprints -> http://www.nmap.org/submit/
   Attempt to guess an unknown -> nmap -O -osscan-guess [target]
   Service version detection -> nmap -sV [target]
   Troubleshooting version scans -> nmap -sV -version-trace [target]
   Perform a RPC scan -> nmap -sR [target]

Timing Options
   Timing Templates -> nmap -T [0-5] [target]
   Set the packet TTL -> nmap -ttl [time] [target]
   Minimum of parallel connections -> nmap -min-parallelism [number] [target]
   Maximum of parallel connection -> nmap -max-parallelism [number] [target]
   Minimum host group size -> nmap -min-hostgroup [number] [targets]
   Maximum host group size -> nmap -max-hostgroup [number] [targets]
   Maximum RTT timeout -> nmap -initial-rtt-timeout [time] [target]
   Initial RTT timeout -> nmap -max-rtt-timeout [TTL] [target]
   Maximum retries -> nmap -max-retries [number] [target]
   Host timeout -> nmap -host-timeout [time] [target]
   Minimum Scan delay -> nmap -scan-delay [time] [target]
   Maximum scan delay -> nmap -max-scan-delay [time] [target]
   Minimum packet rate -> nmap -min-rate [number] [target]
   Maximum packet rate -> nmap -max-rate [number] [target]
   Defeat reset rate limits -> nmap -defeat-rst-ratelimit [target]

Firewall Evasion Techniques
   Fragment packets -> nmap -f [target]
   Specify a specific MTU -> nmap -mtu [MTU] [target]
   Use a decoy -> nmap -D RND: [number] [target]
   Idle zombie scan -> nmap -sI [zombie] [target]
   Manually specify a source port -> nmap -source-port [port] [target]
   Append random data -> nmap -data-length [size] [target]
   Randomize target scan order -> nmap -randomize-hosts [target]
   Spoof MAC Address -> nmap -spoof-mac [MAC|0|vendor] [target]
   Send bad checksums -> nmap -badsum [target]

Output Options
   Save output to a text file -> nmap -oN [scan.txt] [target]
   Save output to a xml file -> nmap -oX [scan.xml] [target]
   Grepable output -> nmap -oG [scan.txt] [target]
   Output all supported file types -> nmap -oA [path/filename] [target]
   Periodically display statistics -> nmap -stats-every [time] [target]
   133t output -> nmap -oS [scan.txt] [target]

Troubleshooting and debugging
```

```
   Help -> nmap -h
   Display Nmap version -> nmap -V
   Verbose output -> nmap -v [target]
   Debugging -> nmap -d [target]
   Display port state reason -> nmap -reason [target]
   Only display open ports -> nmap -open [target]
   Trace packets -> nmap -packet-trace [target]
   Display host networking -> nmap -iflist
   Specify a network interface -> nmap -e [interface] [target]

Nmap Scripting Engine
   Execute individual scripts -> nmap -script [script.nse] [target]
   Execute multiple scripts -> nmap -script [expression] [target]
   Script categories -> all, auth, default, discovery, external, intrusive, malware,
       safe, vuln
   Execute scripts by category -> nmap -script [category] [target]
   Execute multiple scripts categories -> nmap -script [category1,category2, etc]
   Troubleshoot scripts -> nmap -script [script] -script-trace [target]
   Update the script database -> nmap -script-updatedb

Ndiff
   Comparison using Ndiff -> ndiff [scan1.xml] [scan2.xml]
   Ndiff verbose mode -> ndiff -v [scan1.xml] [scan2.xml]
   XML output mode -> ndiff -xml [scan1.xm] [scan2.xml]
```