Experiment No. 8
Download and install nmap. Use it with different options to scan open ports, perform OS fingerprinting, do a ping scan, tcp port scan, udp port scan, etc.
Date of Performance:
Date of Submission:

Experiment No. 8

Title:Download and install nmap. Use it with different options to scan open ports, perform OS fingerprinting, do a ping scan, tcp port scan, udp port scan, etc.

Aim: To use Nmap tool to perform port scan

Objectives:

• To Learn the Namp tool and its features.

Theory:

Nmap (network mapper) is the world's leading network security scanning tool for Linux systems. It helps identify open ports and prevents potential network security threats.

Nmap is an essential network scanning tool due to its accurate, simple-to-use, and flexible interface with many advanced features.

How to Use Nmap to Check Ports

Nmap is a versatile command-line tool that performs powerful port scans. To conduct a simple scan, use the nmap command without any options:

```
nmap [target]
```

The target is either a domain name or an IP address. For example, to scan the website *scanme.nmap.org*, use:

```
nmap scanme.nmap.org
```

```
kb@phoenixNAP:~$ nmap scanme.nmap.org
Starting Nmap 7.80 ( https://nmap.org ) at 2024-04-05 13:45 CEST
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.18s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Not shown: 996 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
9929/tcp  open  nping-echo
31337/tcp open  Elite

Nmap done: 1 IP address (1 host up) scanned in 26.51 seconds
```

The command without any options scans the most common 1000 ports. Nmap can scan a single port, a port range, or all ports on a target. Below are step-by-step instructions on how to use Nmap to scan for open ports in various ways.

Scan a Single Port

To use Nmap to scan a single port on a target, use the following syntax:

```
nmap -p [port] [target]
```

Substitute the placeholders with actual port and target values. For example, to scan port 80 on *scanme.nmap.org*, use:

```
nmap -p 80 scanme.nmap.org
```

```
kb@phoenixNAP:~$ nmap -p 80 scanme.nmap.org
Starting Nmap 7.80 ( https://nmap.org ) at 2024-04-03 15:49 CEST
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.18s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f

PORT   STATE SERVICE
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 0.42 seconds
```

The output shows the port number and protocol (80/tcp), the port's state (open), and the service related to the port (http).

Scan All Ports

To scan all port numbers (1-65535), use the following syntax:

```
nmap -p- [target]
```

For example:

```
nmap -p- scanme.nmap.org
```

```
kb@phoenixNAP:~$ nmap -p- scanme.nmap.org
Starting Nmap 7.80 ( https://nmap.org ) at 2024-04-05 13:17 CEST
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.18s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Not shown: 65531 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
9929/tcp  open  nping-echo
31337/tcp open  Elite

Nmap done: 1 IP address (1 host up) scanned in 933.20 seconds
```

The scan takes time to complete. The command performs a comprehensive scan of all port numbers. It thoroughly assesses the target network and shows open ports for the provided location.

Alternatively, to scan all standard ports, use the fast scan with the `-F` tag:

```
nmap -F [target]
```

For example:

```
nmap -F scanme.nmap.org
```

```
kb@phoenixNAP:~$ nmap -F scanme.nmap.org
Starting Nmap 7.80 ( https://nmap.org ) at 2024-04-05 13:47 CEST
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.18s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Not shown: 98 filtered ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 4.67 seconds
```

The fast scan checks the 100 most common ports. This method reduces the scan time, which is helpful with large networks.

Scan a Series of Ports

The Nmap tool offers several different ways to scan multiple ports. The examples below demonstrate how to use the tool.

To scan a port range, use the `-p` option, the starting and ending port numbers:

```
nmap -p [start]-[end] [target]
```

For example, to scan the first 200 ports, use:

```
nmap -p 1-200 scanme.nmap.org
```

```
kb@phoenixNAP:~$ nmap -p 1-200 scanme.nmap.org
Starting Nmap 7.80 ( https://nmap.org ) at 2024-04-05 13:52 CEST
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.22s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Not shown: 198 filtered ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 6.71 seconds
```

To scan multiple specific ports, use a comma-separated list:

```
nmap -p [port1, port2, etc] [target]
```

For example:

```
nmap -p 22,53,80 scanme.nmap.org
```

```
kb@phoenixNAP:~$ nmap -p 22,53,80 scanme.nmap.org
Starting Nmap 7.80 ( https://nmap.org ) at 2024-04-05 14:01 CEST
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.18s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f

PORT   STATE    SERVICE
22/tcp open     ssh
53/tcp filtered domain
80/tcp open     http

Nmap done: 1 IP address (1 host up) scanned in 2.41 seconds
```

Combine the two methods to scan both specific port numbers and ranges. For example:

```
nmap -p 22,80,100-200 scanme.nmap.org
```

```
kb@phoenixNAP:~$ nmap -p 22,80,100-200 scanme.nmap.org
Starting Nmap 7.80 ( https://nmap.org ) at 2024-04-05 14:03 CEST
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.18s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Not shown: 101 filtered ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 4.84 seconds
```

Scanning targeted ports or a port range shortens the scan time.

Common Ports

There are many standardized ports associated with specific services. Use the list below as a reference for these ports and their related service:

- **21** (FTP). File transfer protocol.
- **22** (SSH). Secure shell.
- **25** (SMTP). Simple mail transfer protocol.
- **53** (DNS). Domain name system.
- **67**, **68** (DHCP). Dynamic host configuration protocol.
- **80** (HTTP). Hypertext transfer protocol.
- **110** (POP3). Post office protocol version 3.
- **123** (NTP). Network time protocol.
- **143** (IMAP). Internet access message protocol.
- **443** (HTTPS). Hypertext transfer protocol secure.
- **465** (SMTPS). SMTP secure.
- **631** (CUPS). Common Unix printing system.
- **993** (IMAPS). IMAP secure.
- **995** (POP3S). POP3 secure.
- **3306** (MySQL). MySQL database server.
- **3389** (RDP). Remote desktop protocol.
- **8080** (HTTP alternate). HTTP alternate, used for proxy servers.

Nmap helps discover port statuses and indicates how to configure a Linux firewall to block traffic on a particular port or allow some traffic.

For example, setting a firewall to block all traffic on port 80 means users won't be able to load any website. Alternatively, firewall rules can be set to allow some traffic to ports.

Use Nmap, a firewall, and other network security tools to scan traffic on a particular port and watch for suspicious activity.

Port States Recognized by Nmap

Nmap divides port status into **six different states**. The possible port states Nmap recognizes are:

- **open**. The service on the associated port is active and listens for incoming connections. The port is available for connections.
- **closed**. No service is listening on the port. No services are bound on the port, and the port will refuse all incoming connections.
- **filtered**. The port state is unknown. The port's status is concealed or restricted due to packet filtering, firewall rules, or a network security device configuration.
- **unfiltered**. The port state is unknown. The port is accessible and unrestricted but has no active service linked to it.
- **open|filtered**. The port state is open or filtered. Nmap cannot determine which due to network conditions.

- **closed|filtered**. The port state is closed or filtered. The exact state is indeterminate due to network conditions.

Implementation:

```
ubuntu@ubuntu:~$ nmap www.google.com
Starting Nmap 7.94 ( https://nmap.org ) at 2024-04-19 00:14 UTC
Nmap scan report for www.google.com (142.251.42.100)
Host is up (0.0071s latency).
Other addresses for www.google.com (not scanned): 2404:6800:4009:800::2004
rDNS record for 142.251.42.100: bom07s45-in-f4.1e100.net
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 4.90 seconds
```

```
Starting Nmap 7.94 ( https://nmap.org ) at 2024-04-19 00:22 UTC
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.30s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2
f
Not shown: 95 closed tcp ports (conn-refused)
PORT    STATE    SERVICE
22/tcp  open     ssh
80/tcp  open     http
135/tcp filtered msrpc
139/tcp filtered netbios-ssn
445/tcp filtered microsoft-ds
```

```
ubuntu@ubuntu:~$ nmap -p 1-10 scanme.nmap.org
Starting Nmap 7.94 ( https://nmap.org ) at 2024-04-19 00:23 UTC
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.32s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2
f

PORT   STATE  SERVICE
1/tcp  closed tcpmux
2/tcp  closed compressnet
3/tcp  closed compressnet
4/tcp  closed unknown
5/tcp  closed rje
6/tcp  closed unknown
7/tcp  closed echo
8/tcp  closed unknown
9/tcp  closed discard
10/tcp closed unknown
```

Conclusion: Comment on use of Nmap tool.

The aim of detecting ARP spoofing using nmap, ARPWATCH, and Wireshark was achieved through comprehensive understanding and practical application of ARP spoofing detection techniques. By meeting the objectives of understanding ARP spoofing and utilizing ARPWATCH, participants gained valuable insights into identifying and mitigating this common network attack. Nmap provided network scanning capabilities to detect unusual ARP traffic patterns, while ARPWATCH served as a dedicated tool for monitoring ARP activity and detecting spoofed ARP packets. Wireshark complemented these tools by enabling detailed packet analysis, facilitating the identification of ARP spoofing attacks through abnormal network behavior.