



IoT Security – Autumn 2024

Lab 7: Port Scanning on IoT Devices

Manh Bui
School of Electrical and Data Engineering
Email: DucManh.Bui@uts.edu.au

Objectives



- This lab provides an overview of **scanning network traffic** the Nmap tool
- The flow of implementation:
 - Part 1: Perform a Nmap network discovery scan
 - Part 2: Compare a Nmap TCP default port scan and a full scan
 - Part 3: Perform a Nmap UDP scan
 - Part 4: Perform Nmap OS and Service Foot Printing

Analysis of cyberattacks on IoT devices



- IoT systems are **vulnerable** to various cyberattacks (Attacks on IoT devices, IoT web apps, IoT servers,...)
- Some known attacks are Denial of Service (**DoS**), Distributed Denial of Service (**DDoS**), Malware, Side-channel,...

How does
cybercrime
identify the
vulnerabilities?

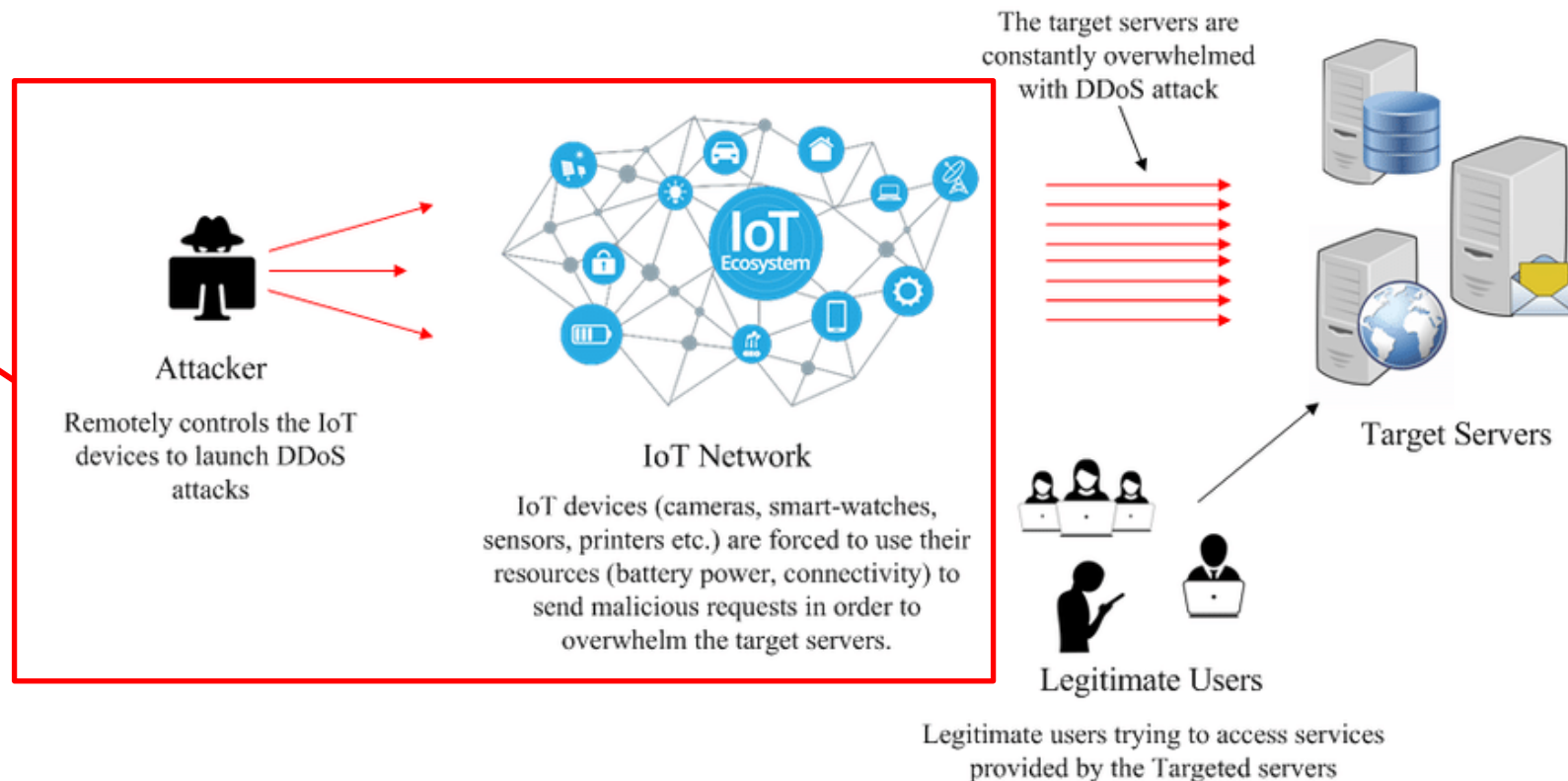


Figure 1: Example of DDoS attacks on IoT environment

Network Scanning



- Network Scanning is the process of **collecting and analysing** machine networking
- It aims to identify servers, end devices, running services, open ports, software versions, and operating systems...
- It can be used for legal and illegal purposes:
 - Legal: Maintaining and enhancing networking infrastructure
 - Illegal: Analyze network vulnerabilities
- Types of scanning techniques: **Port Scanning**, Vulnerability Scanning, IP Scanning, OS Detection,...

What is a Port?



- Port is a virtual point where network connections start and end
- Ports are assigned **port numbers** which, conjunct with an IP address, form vital information that each internet service provider (ISP) **uses to fulfill requests**
- **Port numbers** range from **0 to 65535**
- Ports **numbered 0 to 1023** are “**well-known**” **ports**, which are typically reserved for **internet usage**
- Some of the most frequently used ports:
 - Port 20 (UDP): File Transfer Protocol (FTP) used for transferring data
 - Port 22 (TCP): Secure Shell (SSH) protocol for FTP, port forwarding and secure logins
 - Port 23 (TCP): Telnet protocol used for unencrypted communication
 - Port 53 (UDP): The DNS which translates the internet domain to machine-readable IP
 - Port 80 (TCP): The World Wide Web HTTP

Port Scanning



- A port scan is a common technique hackers use to **discover open doors** or **weak points in a network**
- A port scan attack helps cyber criminals **find open ports** and **figure out whether they are receiving or sending data**
- Some port scanning methods: Ping scans, Vanilla scan, **SYN scan**, **UDP scan**, XMAS and FIN scans, FTP bounce scan, Sweep scan
- Tools for port scanning: Wireshark, **Nmap**,...

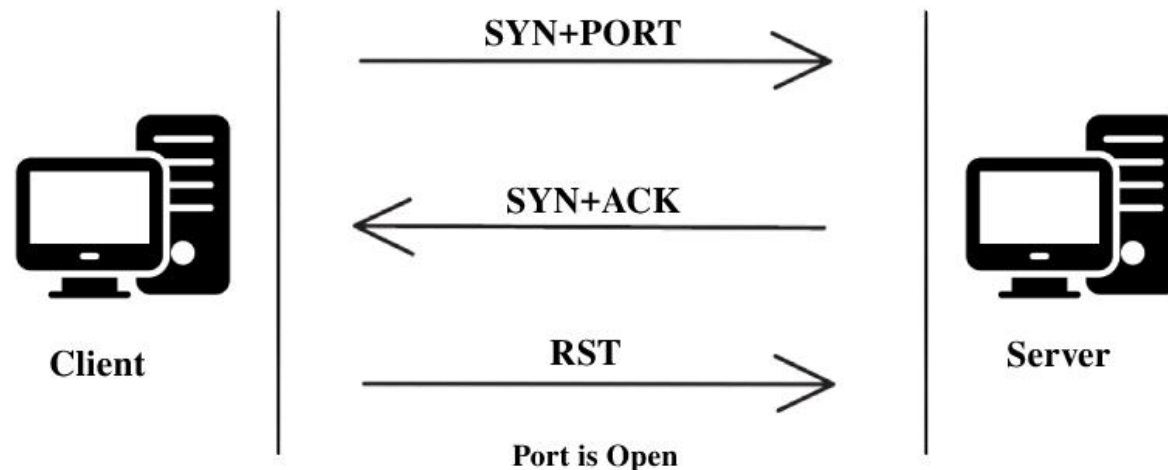


Figure 2: Example of SYN scan

Nmap tool



- Nmap, or “Network mapper,” is a free and open source network port scanner.

- Nmap provides information on
 - Every active IP
 - Your network as whole
 - Vulnerabilities

- Nmap is used to scan
 - Enterprise-scale networks
 - Small business networks
 - Connected devices
 - IoT device and traffic

- Nmap common functions
 - Ping scanning
 - Port scanning
 - Host/OS scanning
 - Scan top ports
 - Output to files
 - Disable DNS resolution

Required Resources



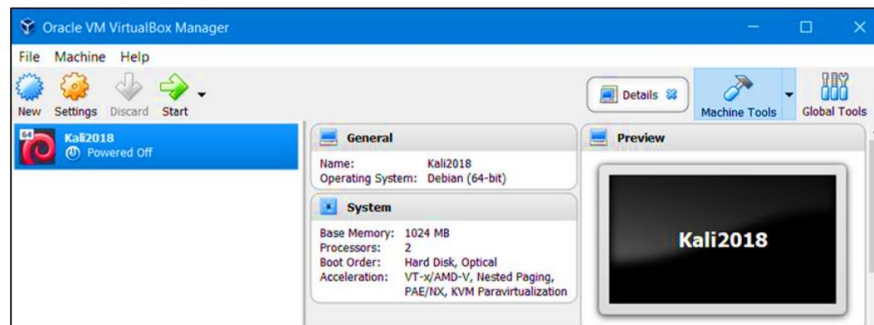
Part1: Raspberry Pi 3 Model B or later



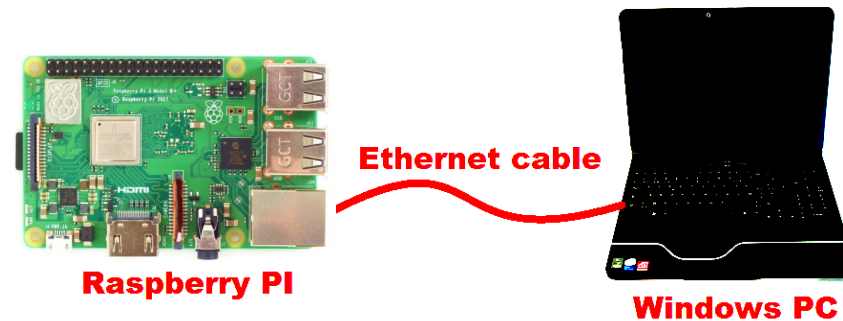
Part2: 8GB Micro SD card (minimum required)



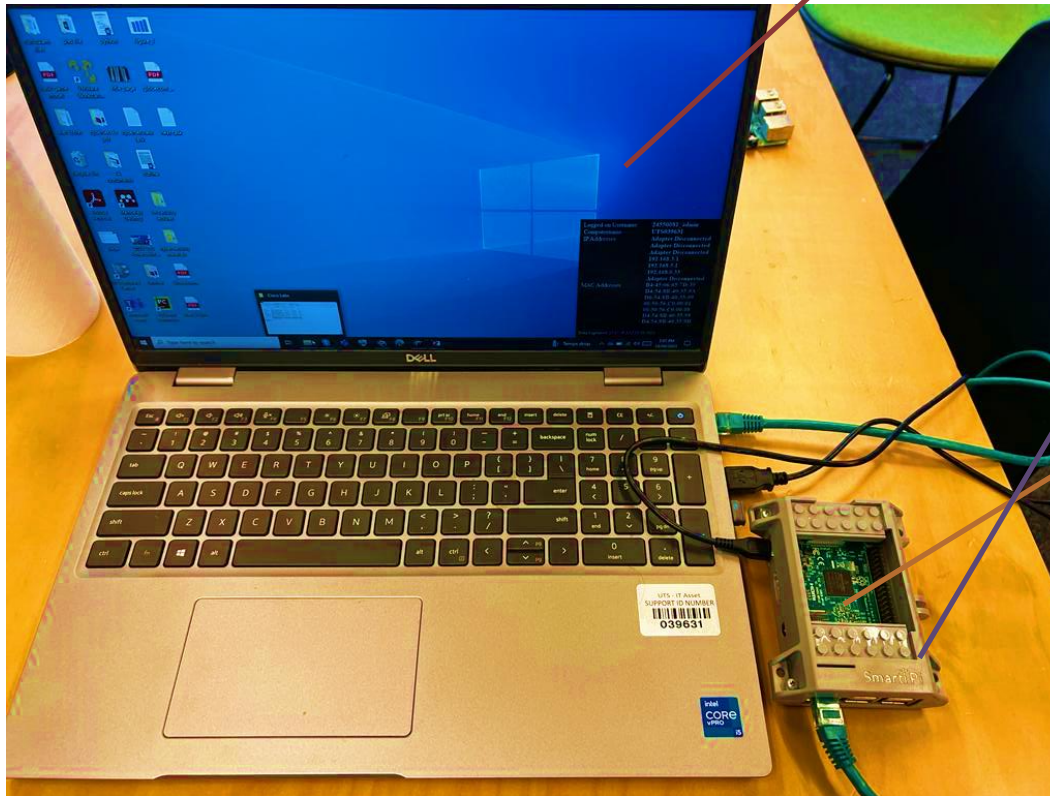
Part3: PC with IoTSec Kali VM



Part4: Network connectivity between PC and Raspberry Pi



Environment Setup



PC with IoTSec Kali
VM

Sd card inserted

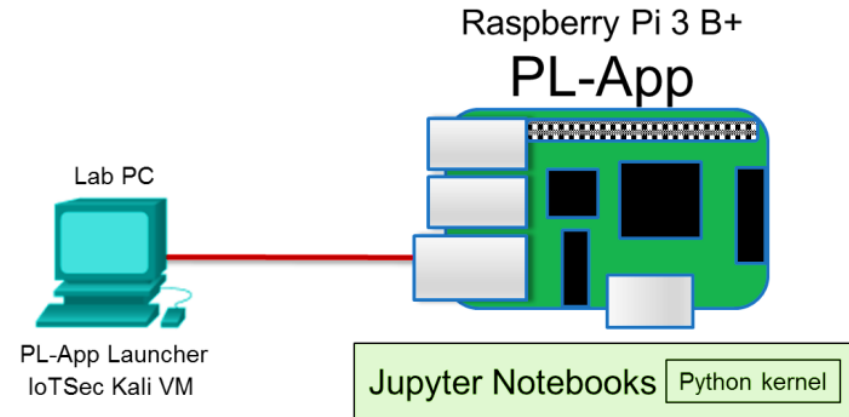
Raspberry Pi 3

Part 1: Perform a Nmap Network Discovery Scan

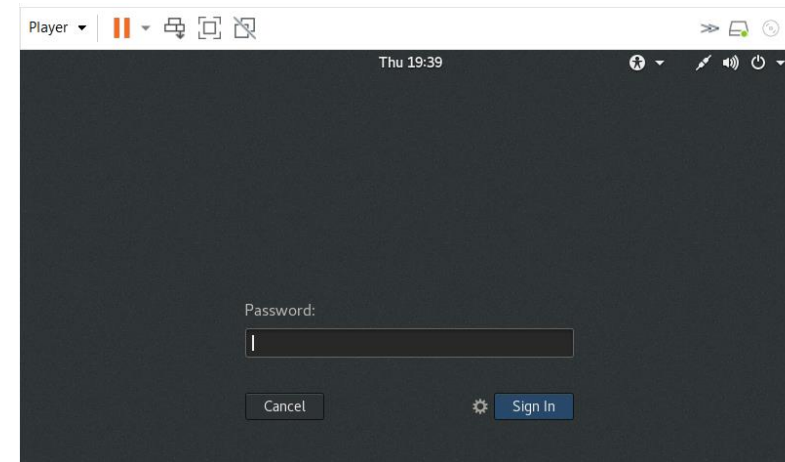


Step 1: Use Kali to perform a host discovery scan

a. Set up the topology by connecting the Raspberry Pi to the PC.



b. Start the IoTSec Kali VM and log in.



Part 1: Perform a Nmap Network Discovery Scan



a. Set up the topology by connecting the Raspberry Pi to the PC.

Cisco PL-App Launcher

Setup a New DeviceAvailable Devices

| Device Name | Status |
|-------------------------|---------|
| arifRPi1 (203.0.113.20) | Connect |

☐ Use Broadcast mDNS

Add Device Name

Version 1.5.11

io | Sec_Kali [Kunming] - Oracle VM VirtualBox

FileMachineViewInputDevicesHelp

ApplicationsPlacesTerminalTue 20:05

root@kali: ~

FileEditViewSearchTerminalHelp

root@kali:~# ping 203.0.113.20

PING 203.0.113.20 (203.0.113.20) 56(84) bytes of data:

64 bytes from 203.0.113.20: icmp_seq=1 ttl=64 time=1.46 ms

64 bytes from 203.0.113.20: icmp_seq=2 ttl=64 time=1.75 ms

64 bytes from 203.0.113.20: icmp_seq=3 ttl=64 time=1.77 ms

64 bytes from 203.0.113.20: icmp_seq=4 ttl=64 time=1.49 ms

64 bytes from 203.0.113.20: icmp_seq=5 ttl=64 time=1.60 ms

64 bytes from 203.0.113.20: icmp_seq=6 ttl=64 time=1.31 ms

64 bytes from 203.0.113.20: icmp_seq=7 ttl=64 time=3.35 ms

64 bytes from 203.0.113.20: icmp_seq=8 ttl=64 time=1.86 ms

64 bytes from 203.0.113.20: icmp_seq=9 ttl=64 time=1.68 ms

64 bytes from 203.0.113.20: icmp_seq=10 ttl=64 time=1.69 ms

64 bytes from 203.0.113.20: icmp_seq=11 ttl=64 time=1.24 ms

64 bytes from 203.0.113.20: icmp_seq=12 ttl=64 time=1.23 ms

64 bytes from 203.0.113.20: icmp_seq=13 ttl=64 time=1.75 ms

64 bytes from 203.0.113.20: icmp_seq=14 ttl=64 time=1.70 ms

64 bytes from 203.0.113.20: icmp_seq=15 ttl=64 time=1.48 ms

64 bytes from 203.0.113.20: icmp_seq=16 ttl=64 time=1.68 ms

64 bytes from 203.0.113.20: icmp_seq=17 ttl=64 time=1.73 ms

Not secure | 203.0.113.20/tree?

CPU:Memory:Local | Up Time:arifRPi1Logout

FilesRunning

Select items to perform actions on them.

UploadNew

0 /

NameLast ModifiedFile size

Course Materials

4 years ago

myfiles

4 years ago

READMEFIRST.ipynb

6 years ago

7.66 kB

Part 1: Perform a Nmap Network Discovery Scan



c. Open a terminal and start the DHCP server on the Kali VM.

```
root@kali:~# lab_support_files/scripts/start_dhcp.sh
```

```
root@kali: ~
File Edit View Search Terminal Help
root@kali:~# lab_support_files/scripts/start_dhcp.sh
[ ok ] Starting isc-dhcp-server (via systemctl): isc-dhcp-server.service.
root@kali:~#
```

d. Verify that Kali VM is assigned an IP address on eth0.

```
root@kali:~# ifconfig
```

```
root@kali: ~
File Edit View Search Terminal Help
root@kali:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 203.0.113.1 netmask 255.255.255.0 broadcast 203.0.113.255
    inet6 fe80::a00:27ff:fe6a:deae prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:6a:de:ae txqueuelen 1000 (Ethernet)
    RX packets 894 bytes 74040 (72.3 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 45 bytes 4128 (4.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 33 bytes 2440 (2.3 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 33 bytes 2440 (2.3 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Part 1: Perform a Nmap Network Discovery Scan



e. Open the man page for nmap in Kali VM and review the options that are available in Nmap.

root@kali:~# **man nmap**

```
root@kali: ~
File Edit View Search Terminal Help
root@kali:~# man nmap
```

```
root@kali: ~
File Edit View Search Terminal Help
even included here.

Nmap 7.60 ( 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; 10.0.0-255.1-255

-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/PY/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given
ports
-PE/PP/PM: ICMP echo, timestamp, and netmask request discovery prob
es
-P0[protocol list]: IP Protocol Ping
-n/-R: Never do DNS resolution/Always resolve [default: sometimes]
--dns-servers <serv1[,serv2],...>: Specify custom DNS servers
Manual page nmap(1) line 92 (press h for help or q to quit)
```


Part 1: Perform a Nmap Network Discovery Scan



f. Perform a scan on your network by specifying the network address and bit mask.

```
root@kali:~# nmap 203.0.113.0/24
```

```
root@kali: ~
File Edit View Search Terminal Help

root@kali:~# man nmap
root@kali:~# nmap 203.0.113.0/24

Starting Nmap 7.60 ( https://nmap.org ) at 2024-03-12 20:07 EDT
Nmap scan report for 203.0.113.19
Host is up (0.00029s latency).
Not shown: 992 filtered ports
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
902/tcp    open  iss-realsecure
912/tcp    open  apex-mesh
2701/tcp   open  sms-rcinfo
3389/tcp   open  ms-wbt-server
5357/tcp   open  wsdapi
MAC Address: B4:45:06:65:7D:39 (Unknown)

Nmap scan report for 203.0.113.20
Host is up (0.00087s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
80/tcp    open  http
MAC Address: B8:27:EB:6D:7A:55 (Raspberry Pi Foundation)

Nmap scan report for 203.0.113.1
Host is up (0.0000010s latency).
All 1000 scanned ports on 203.0.113.1 are closed

Nmap done: 256 IP addresses (3 hosts up) scanned in 32.06 seconds
root@kali:~#
```

Part 1: Perform a Nmap Network Discovery Scan



g. Sometimes a device would not reply to Nmap's initial network discovery scan because of a firewall, IDS/IPS system etc. Instead of relying on the initial scan to discover hosts that are alive for further scanning.

we can use Nmap to scan the network by assuming all hosts are alive.

root@kali:~# **nmap -Pn 203.0.113.0/24** – answer will vary

```
root@kali: ~
File Edit View Search Terminal Help
root@kali:~# nmap -Pn 203.0.113.0/24

Starting Nmap 7.60 ( https://nmap.org ) at 2024-03-12 20:15 EDT
Nmap scan report for 203.0.113.19
Host is up (0.00043s latency).
Not shown: 992 filtered ports
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
902/tcp    open  iss-realservice
912/tcp    open  apex-mesh
2701/tcp   open  sms-rcinfo
3389/tcp   open  ms-wbt-server
5357/tcp   open  wsdaapi
MAC Address: B4:45:06:65:7D:39 (Unknown)

Nmap scan report for 203.0.113.20
Host is up (0.00087s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
80/tcp    open  http
MAC Address: B8:27:EB:6D:7A:55 (Raspberry Pi Foundation)

Nmap scan report for 203.0.113.1
Host is up (0.0000020s latency).
All 1000 scanned ports on 203.0.113.1 are closed

Nmap done: 256 IP addresses (3 hosts up) scanned in 38.70 seconds
root@kali:~#
```

Part 2: Compare a Nmap TCP Default Scan and Full Scan



Step 1: Using Wireshark to display Nmap scans

a. In a Kali VM terminal, start Wireshark. Wireshark is used to monitor the traffic while scanning the network using Nmap. Click **OK** for the warning message regarding running Wireshark as a root user.

```
root@kali:~# wireshark
```

b. Select the **eth0** interface and click **Capture** to start capturing packets

The screenshot displays a Kali Linux terminal window and the Wireshark network protocol analyzer. The terminal window shows the execution of Nmap 7.60, which has scanned the host 203.0.113.19. The output indicates that the host is up and lists several open ports (135/tcp, 139/tcp, 445/tcp, 902/tcp, 912/tcp, 2701/tcp, 3389/tcp, 5357/tcp). The Wireshark interface is open, showing a list of captured packets. The first packet is a TCP scan from 203.0.113.19 to 8.8.8.8. The packet details pane shows the frame structure: Ethernet II, Internet Protocol Version 4, and Transmission Control Protocol. The packet list pane shows the details of the captured packets, including the source and destination IP addresses, ports, and protocols.

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|-------------|--------------|-------------|----------|--------|------|
| 1 | 0.000000000 | 203.0.113.19 | 8.8.8.8 | TCP | 66 | 561 |
| 2 | 0.00013086 | 203.0.113.19 | 8.8.8.8 | TCP | 66 | 561 |
| 3 | 0.003824730 | 203.0.113.19 | 8.8.8.8 | TCP | 66 | 561 |
| 4 | 0.003827020 | 203.0.113.19 | 8.8.8.8 | TCP | 66 | 561 |
| 5 | 0.014532333 | 203.0.113.19 | 8.8.8.8 | TCP | 66 | 561 |
| 6 | 0.014545850 | 203.0.113.19 | 8.8.8.8 | TCP | 66 | 561 |

Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
Ethernet II, Src: b4:45:06:65:7d:39 (b4:45:06:65:7d:39), Dst: PcsCompu_6a:de:ae (b4:45:06:65:7d:39)
Internet Protocol Version 4, Src: 203.0.113.19, Dst: 8.8.8.8
Transmission Control Protocol, Src Port: 56168, Dst Port: 53, Seq: 0, Len: 0

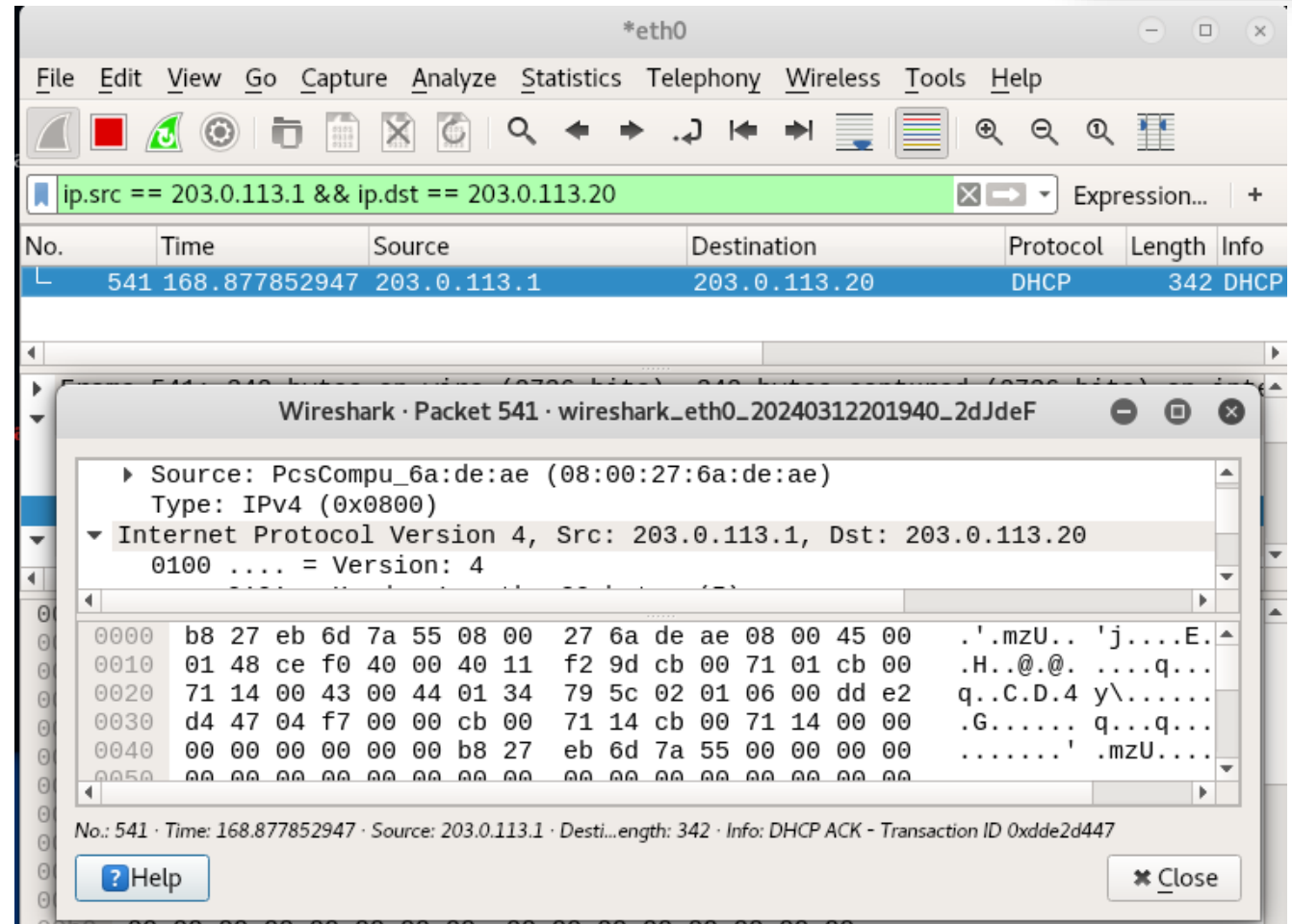
| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|-------------|-------------------|-------------------|----------|--------|----------------------------|
| 13 | 0.501728778 | 203.0.113.20 | 8.8.8.8 | DNS | 81 | Standard query response |
| 14 | 0.746765920 | 203.0.113.19 | 8.8.8.8 | DNS | 94 | Standard query response |
| 15 | 0.820081403 | 203.0.113.19 | 8.8.8.8 | DNS | 90 | Standard query response |
| 16 | 1.010928562 | b4:45:06:65:7d:39 | PcsCompu_6a:de:ae | ARP | 60 | Who has 08:00:27:6a:de:ae? |
| 17 | 1.010954474 | PcsCompu_6a:de:ae | b4:45:06:65:7d:39 | ARP | 42 | 203.0.113.19 |
| 18 | 1.014824715 | 203.0.113.19 | 8.8.8.8 | TCP | 66 | [TCP] |
| 19 | 1.014830990 | 203.0.113.19 | 8.8.8.8 | TCP | 66 | [TCP] |
| 20 | 1.032474677 | 203.0.113.19 | 8.8.8.8 | DNS | 81 | Standard query response |
| 21 | 1.183647058 | 203.0.113.19 | 8.8.8.8 | DNS | 94 | Standard query response |

Part 2: Compare a Nmap TCP Default Scan and Full Scan



c. There can be a lot of traffic on the network. A display filter is applied to limit the number of captured packets displayed to just those that you are interested in. The interesting traffic is between the IP address of the Kali VM and the Raspberry Pi. Replace IP address of the Raspberry Pi with the IP address. Apply the following display filter in Wireshark using IP address of Kali VM as the source address and IP address of your Raspberry Pi as the destination address.

`ip.src == 203.0.113.1 && ip.dst == 203.0.113.20`



Part 2: Compare a Nmap TCP Default Scan and Full Scan-Cont.



Step 2: Nmap TCP default scan

- a. In the terminal, enter the following command to start the Nmap scanning.
root@kali:~# **nmap 203.0.113.0/24**

```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# nmap 203.0.113.0/24  
Starting Nmap 7.60 ( https://nmap.org ) at 2024-03-12 20:30 EDT  
Nmap scan report for 203.0.113.19  
Host is up (0.00034s latency).  
Not shown: 992 filtered ports  
PORT      STATE SERVICE  
135/tcp    open  msrpc  
139/tcp    open  netbios-ssn  
445/tcp    open  microsoft-ds  
902/tcp    open  iss-realsecure  
912/tcp    open  apex-mesh  
2701/tcp   open  sms-rcinfo  
3389/tcp   open  ms-wbt-server  
5357/tcp   open  wsdapi  
MAC Address: B4:45:06:65:7D:39 (Unknown)  
  
Nmap scan report for 203.0.113.20  
Host is up (0.00094s latency).  
Not shown: 999 closed ports  
PORT      STATE SERVICE  
80/tcp    open  http  
MAC Address: B8:27:EB:6D:7A:55 (Raspberry Pi Foundation)  
  
Nmap scan report for 203.0.113.1  
Host is up (0.0000010s latency).  
All 1000 scanned ports on 203.0.113.1 are closed  
  
Nmap done: 256 IP addresses (3 hosts up) scanned in 32.46 seconds
```

Part 2: Compare a Nmap TCP Default Scan and Full Scan-Cont.



After Nmap reports the result of the scan, stop the Wireshark capture. Click the arrow next to the display filter field to filter the results of the scan. Review the Wireshark output. Which TCP flag is Nmap using to discover the open ports?

Review the Wireshark output. Which TCP flag is Nmap using to discover the open ports?

SYN

Notice the ports that are being tested. The default Nmap scan does not test all ports. How many ports does a default Nmap scan test? (Do a web search if necessary.)

Only 1,000 of the most common ports.

Look at the results of the nmap scan in the terminal. What ports are identified?

Answers may vary. Ports 22 and 80 should be open, but others may also be found.

| No. | Time | Destination | Protocol | Length | Info |
|-----|----------|--------------|----------|--------|--|
| 1 | 0.000000 | 203.0.113.20 | DHCP | 342 | DHCP ACK - Transaction ID 0xdd2d447 |
| 2 | 0.000000 | 203.0.113.20 | DHCP | 342 | DHCP ACK - Transaction ID 0xa79fa338 |
| 3 | 0.000000 | 203.0.113.20 | TCP | 58 | 64919 → 21 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 4 | 0.000000 | 203.0.113.20 | TCP | 58 | 64919 → 995 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 5 | 0.000000 | 203.0.113.20 | TCP | 58 | 64919 → 143 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 6 | 0.000000 | 203.0.113.20 | TCP | 58 | 64919 → 111 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 7 | 0.000000 | 203.0.113.20 | TCP | 58 | 64919 → 1025 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 8 | 0.000000 | 203.0.113.20 | TCP | 58 | 64919 → 5900 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 9 | 0.000000 | 203.0.113.20 | TCP | 58 | 64919 → 3306 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 10 | 0.000000 | 203.0.113.20 | TCP | 58 | 64919 → 199 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 11 | 0.000000 | 203.0.113.20 | TCP | 58 | 64919 → 587 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 12 | 0.000000 | 203.0.113.20 | TCP | 58 | 64919 → 23 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| 13 | 0.000000 | 203.0.113.20 | TCP | 58 | 64919 → 25 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |

Destination: Raspberr_6d:7a:55 (b8:27:eb:6d:7a:55)
Source: PcsCompu_6a:de:ae (08:00:27:6a:de:ae)
Type: IPv4 (0x0800)

| Offset | Hex | ASCII |
|--------|---|------------------|
| 0000 | b8 27 eb 6d 7a 55 08 00 27 6a de ae 08 00 45 00 | ..mzU.. 'j...E. |
| 0010 | 01 48 ce f0 40 00 40 11 f2 9d cb 00 71 01 cb 00 | .H..@.q... |
| 0020 | 71 14 00 43 00 44 01 34 79 5c 02 01 06 00 dd e2 | q..C.D.4 y\..... |
| 0030 | d4 47 04 f7 00 00 cb 00 71 14 cb 00 71 14 00 00 | .G..... q...q... |
| 0040 | 00 00 00 00 00 00 b8 27 eb 6d 7a 55 00 00 00 00 |' .mzU.... |

Part 2: Compare a Nmap TCP Default Scan and Full Scan-Cont.



Step 3: Nmap TCP full TCP port scan

- Nmap by default only scan a limited number of TCP ports. We would like to scan all 65535 TCP ports.
- Start a new Wireshark capture. Click **Continue without Saving** when prompted to save the capture. to start a new capture.
- Enter the nmap command to scan all the TCP ports in Kali VM.

root@kali:~# **nmap -p 1-65535 203.0.113.20**

```
root@kali:~# nmap -p 1-65535 203.0.113.20

Starting Nmap 7.60 ( https://nmap.org ) at 2024-03-12 20:38 EDT
Nmap scan report for 203.0.113.20
Host is up (0.00067s latency).
Not shown: 65534 closed ports
PORT      STATE SERVICE
80/tcp    open  http
MAC Address: B8:27:EB:6D:7A:55 (Raspberry Pi Foundation)

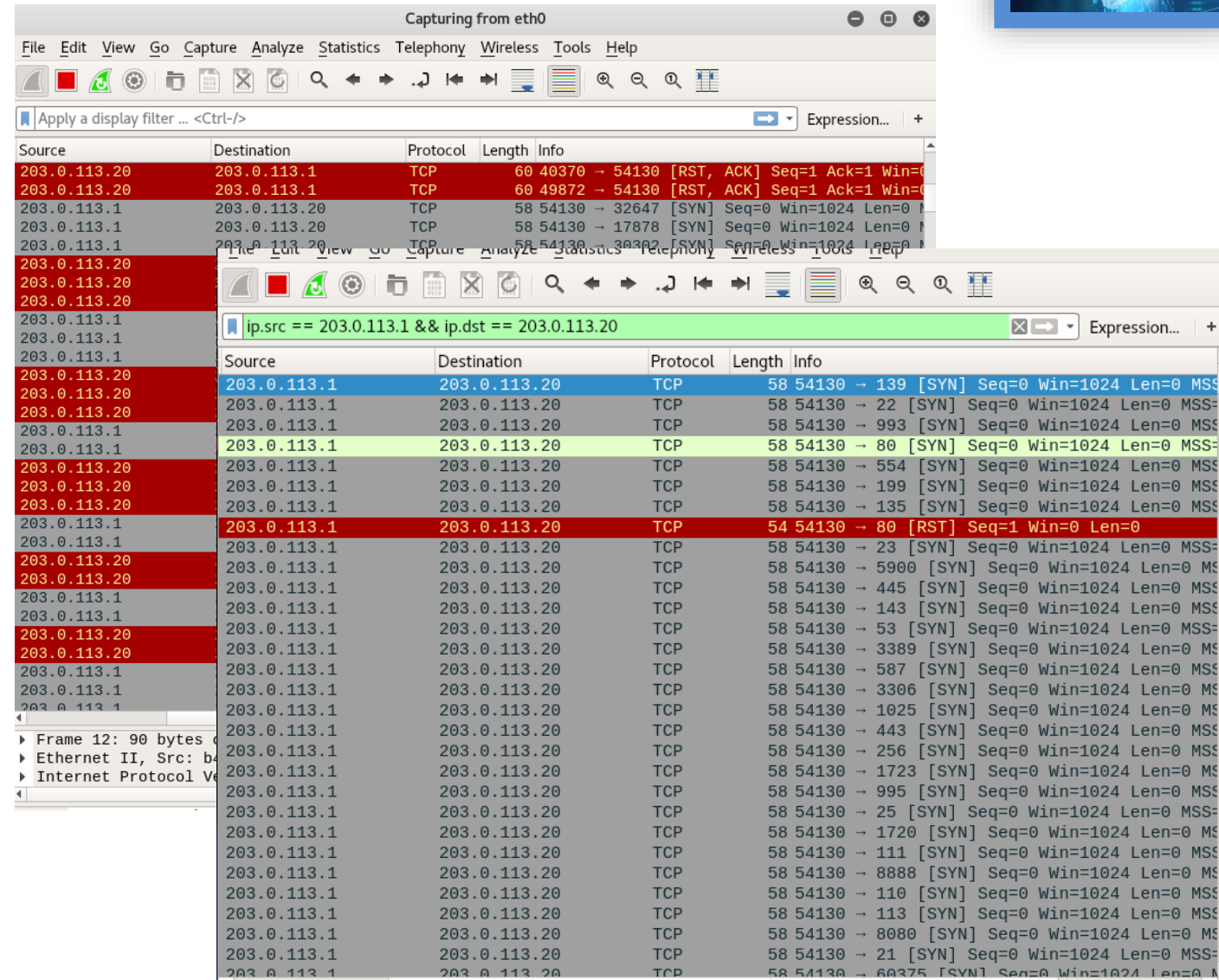
Nmap done: 1 IP address (1 host up) scanned in 29.45 seconds
root@kali:~#
```

Part 2: Compare a Nmap TCP Default Scan and Full Scan-Cont.



d. Watch the Wireshark capture screen. Notice you are sending TCP packets just as before, but the number of ports being scanned has increased.

e. Stop the Wireshark capture when finished and clear the filter.



Part 3: Perform a Nmap UDP scan-Cont.



UDP Header Format

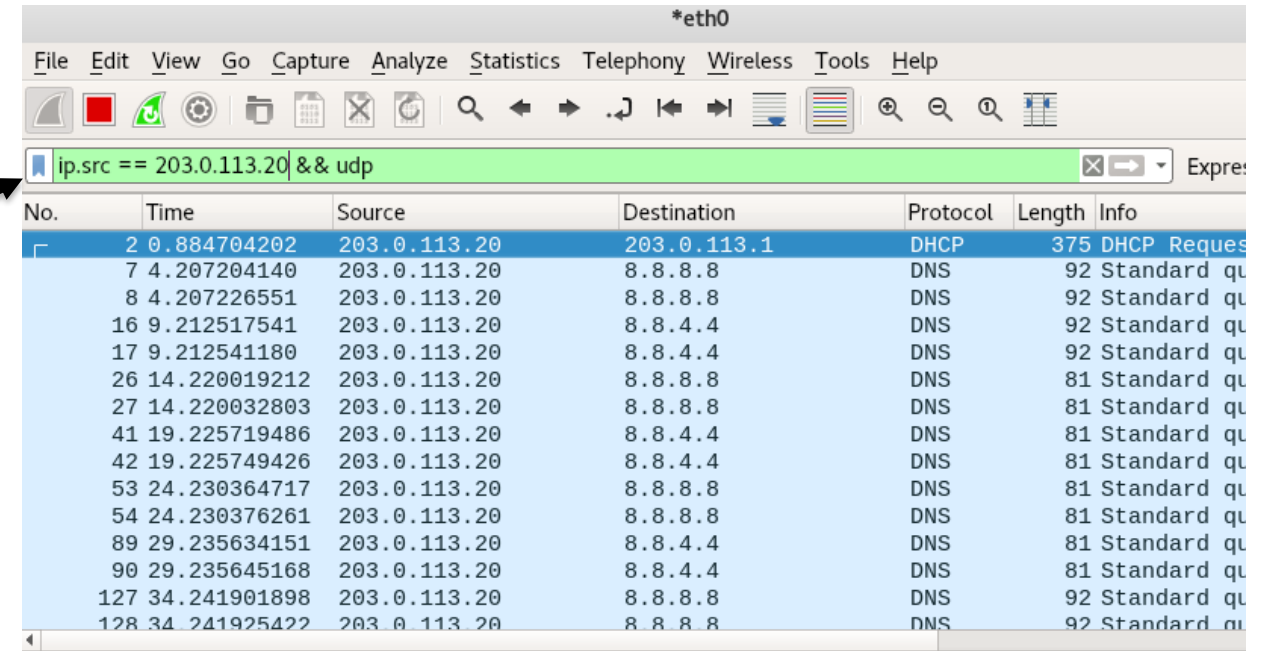
| Source Port | Destination Port |
|-------------|------------------|
| Length | Checksum |

Step 1: UDP scan with a new Wireshark filter.

a. Start a new Wireshark capture. Click Continue without Saving when prompted to save the capture.

b. Apply the following filter in Wireshark:

`ip.src == 203.0.113.20 && udp`



This will allow us to see the UDP traffic generated by nmap to the Raspberry PI. Notice in the UDP header above there are no flags. We can only send UDP packets and receive a possible “port unreachable or destination unreachable” message meaning the port is closed.

Part 3: Perform a Nmap UDP scan-Cont.



c. In the Kali VM terminal, enter the command nmap with -sU option scanning the IP address of your Raspberry Pi.

root@kail:~# nmap -sU -F 203.0.113.13

```
root@kali:~# nmap -sU -F 203.0.113.20
Starting Nmap 7.60 ( https://nmap.org ) at 2024-03-12 20:52 EDT
Nmap scan report for 203.0.113.20
Host is up (0.0016s latency).
Not shown: 98 closed ports
PORT      STATE      SERVICE
68/udp    open|filtered dhcpc
5353/udp  open|filtered zeroconf
MAC Address: B8:27:EB:6D:7A:55 (Raspberry Pi Foundation)
Nmap done: 1 IP address (1 host up) scanned in 114.96 seconds
```

d. It will take a few minutes, but look at the Wireshark capture. As you scroll down the packets, you will see some “Destination unreachable” on different ports indicating the port is closed.

| | Source | Destination | Protocol | Length | Info |
|----|--------------|-------------|----------|--------|--|
| 10 | 203.0.113.20 | 203.0.113.1 | ICMP | 70 | Destination unreachable (Port unreachable) |
| 34 | 203.0.113.20 | 203.0.113.1 | ICMP | 94 | Destination unreachable (Port unreachable) |
| 13 | 203.0.113.20 | 8.8.8.8 | DNS | 81 | Standard query 0x5b97 A 3.debian.pool.nt |
| 43 | 203.0.113.20 | 8.8.8.8 | DNS | 81 | Standard query 0x6797 AAAA 3.debian.pool |
| 52 | 203.0.113.20 | 203.0.113.1 | ICMP | 70 | Destination unreachable (Port unreachable) |
| 22 | 203.0.113.20 | 203.0.113.1 | ICMP | 70 | Destination unreachable (Port unreachable) |
| 52 | 203.0.113.20 | 203.0.113.1 | ICMP | 70 | Destination unreachable (Port unreachable) |
| 60 | 203.0.113.20 | 203.0.113.1 | ICMP | 70 | Destination unreachable (Port unreachable) |
| 51 | 203.0.113.20 | 8.8.4.4 | DNS | 81 | Standard query 0x5b07 A 3.debian.pool.nt |

Part 3: Perform a Nmap UDP scan-Cont.



How many UDP ports are there? How many UDP ports does Nmap scan, by default? (Use web search as necessary.)

There are 65,535 UDP ports. Nmap scans only the most commonly used 1,000.

What are the UDP ports that are open from the scan? What protocols use these UDP ports?

Answers may vary. Ports 68 and 5353 are most likely. The bootp and Multicast DNS use these ports. These services are used by PL-App.

What is the meaning of the -F option? Try the same scan without the -F. What is the difference?

The -F option stands for fast. Only the most common 100 ports are scanned. This makes the scan much faster.

Part 3: Perform a Nmap UDP scan-Cont.



| Apply a display filter ... <Ctrl-/> | | | | | Expression... | + |
|-------------------------------------|--------------|----------|--------|--|---------------|---|
| | Destination | Protocol | Length | Info | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40339 → 5632 Len=0 | | |
| 113.1 | 203.0.113.20 | ISAKMP | 234 | Identity Protection (Main Mode) | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40339 → 68 Len=0 | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40339 → 3703 Len=0 | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40339 → 5000 Len=0 | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40339 → 158 Len=0 | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40339 → 49 Len=0 | | |
| 113.1 | 203.0.113.20 | UDP | 72 | 40339 → 626 Len=30 | | |
| 113.1 | 203.0.113.20 | UDP | 72 | 40340 → 626 Len=30 | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40340 → 49 Len=0 | | |
| 113.20 | 203.0.113.1 | ICMP | 100 | Destination unreachable (Port unreachable) | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40340 → 158 Len=0 | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40340 → 5000 Len=0 | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40340 → 3703 Len=0 | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40340 → 68 Len=0 | | |
| 113.1 | 203.0.113.20 | ISAKMP | 234 | Identity Protection (Main Mode) | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40340 → 5632 Len=0 | | |
| 113.1 | 203.0.113.20 | MDNS | 88 | Standard query 0x0000 PTR _services._dns-sd._udp | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40340 → 67 Len=0 | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40340 → 49193 Len=0 | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40340 → 1023 Len=0 | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40340 → 162 Len=0 | | |
| 113.1 | 203.0.113.20 | UDP | 42 | 40340 → 49194 Len=0 | | |

e. Stop the Wireshark capture when finished and clear the filter.

Part 4: Perform Nmap OS and Service Foot Printing



Step 1: Use Nmap to find a device operating system.

- Start a new Wireshark capture. Click Continue without Saving when prompted to save the capture.
- Apply the following filter in Wireshark using the IP address of Kali VM as the source address and the IP address of your Raspberry Pi as the destination address.

`ip.src == 203.0.113.1 && ip.dst == 203.0.113.20/28`

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|--------------|-------------|--------------|----------|--------|----------------------|
| 168 | 48.280418074 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 47937 -> 1720 [SYN] |
| 169 | 48.280722183 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 47937 -> 53 [SYN] Se |
| 170 | 48.280870770 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 47937 -> 1723 [SYN] |
| 171 | 48.281010084 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 47937 -> 143 [SYN] S |
| 172 | 48.281243884 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 47937 -> 443 [SYN] S |
| 176 | 48.282083347 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 47937 -> 110 [SYN] S |
| 177 | 48.282246245 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 47937 -> 23 [SYN] Se |
| 180 | 48.282468079 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 47937 -> 199 [SYN] S |
| 181 | 48.282607762 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 47937 -> 22 [SYN] Se |
| 182 | 48.282752390 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 47937 -> 445 [SYN] S |
| 188 | 48.284160705 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 47937 -> 2206 [SYN] |

Part 4: Perform Nmap OS and Service Foot Printing-cont



- c. In the Kali VM terminal, enter the command nmap using the -O option scanning the IP address of your Raspberry Pi.

```
root@kali:~# nmap -O 203.0.113.20
```

What operating system did Nmap guess?

```
root@kali:~# nmap -O 203.0.113.20
Starting Nmap 7.60 ( https://nmap.org ) at 2024-03-12 21:01 EDT
Nmap scan report for 203.0.113.20
Host is up (0.0016s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
80/tcp    open  http
MAC Address: B8:27:EB:6D:7A:55 (Raspberry Pi Foundation)
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/).
TCP/IP fingerprint: 280870770 203.0.113.1 203.0.113.20
OS:SCAN(V=7.60%E=4%D=3/12%OT=80%CT=1%CU=38585%PV=N%DS=1%DC=D%G=Y%M=B827EB%TP
OS:M=65F0FAF3%P=x86_64-pc-linux-gnu)SEQ(SP=105%GCD=1%ISR=108%TI=Z%CI=Z%II=IP
OS:%TS=A)OPS(O1=M5B4ST11NW7%O2=M5B4ST11NW7%O3=M5B4NNT11NW7%O4=M5B4ST11NW7%O
OS:5=M5B4ST11NW7%O6=M5B4ST11)WIN(W1=FE88%W2=FE88%W3=FE88%W4=FE88%W5=FE88%W6P
OS:=FE88)ECN(R=Y%DF=Y%T=40%W=FAF0%O=M5B4NNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=0P
OS:%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=P
OS:0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0P
OS:S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(P
OS:R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=P
OS:N%T=40%CD=S) 0 48.284498112 203.0.113.1 203.0.113.20 TCP 58 47937 -> 1726
191 48.284638001 203.0.113.1 203.0.113.20 TCP 58 47937 -> 111
Network Distance: 41 hop 310456 203.0.113.1 203.0.113.20 TCP 58 47937 -> 554
194 48.285041458 203.0.113.1 203.0.113.20 TCP 58 47937 -> 587
OS detection performed. Please report any incorrect results at https://nmap.org/submit/47937 -> 8080
Nmap done: 1 IP address (1 host up) scanned in 24.992 seconds 3.20 TCP 58 47937 -> 80
```

Part 4: Perform Nmap OS and Service Foot Printing-cont



Look at the packets in Wireshark. What protocols were used to determine the OS?

| Destination | Protocol | Length | Info |
|--------------|----------|--------|---|
| 203.0.113.20 | TCP | 70 | [TCP Port numbers reused] 39532 → 80 [SYN] Seq=0 Win=5 |
| 203.0.113.20 | TCP | 54 | 39532 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | ICMP | 162 | Echo (ping) request id=0x95e2, seq=295/998 |
| 203.0.113.20 | ICMP | 192 | Echo (ping) request id=0x95e3, seq=296/1024 |
| 203.0.113.20 | UDP | 342 | 39465 → 38585 Len=300 |
| 203.0.113.20 | TCP | 66 | [TCP Port numbers reused] 39539 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39539 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 74 | 39541 → 80 [<None>] Seq=2295984856 Win=131072 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39542 → 80 [FIN, SYN, PSH, URG] Seq=2295984856 Win=131072 Len=0 |
| 203.0.113.20 | TCP | 74 | 39543 → 80 [ACK] Seq=2295984856 Ack=4154496 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39544 → 1 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 74 | 39545 → 1 [ACK] Seq=2295984856 Ack=4154496 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Retransmission] 39546 → 1 [FIN, PSH, URG] Seq=2295984856 Win=131072 Len=0 |
| 203.0.113.20 | TCP | 74 | 39541 → 80 [<None>] Seq=2295984856 Win=131072 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Retransmission] 39542 → 80 [FIN, SYN, PSH, URG] Seq=2295984856 Win=131072 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39527 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39527 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39528 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39528 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39529 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39529 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 70 | [TCP Port numbers reused] 39530 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39530 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39531 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39531 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 70 | [TCP Port numbers reused] 39532 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39532 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | ICMP | 162 | Echo (ping) request id=0x08c8, seq=295/9985, ttl=58 (|
| 203.0.113.20 | ICMP | 192 | Echo (ping) request id=0x08c9, seq=296/10241, ttl=58 (|
| 203.0.113.20 | UDP | 342 | 39465 → 38585 Len=300 |
| 203.0.113.20 | TCP | 66 | [TCP Port numbers reused] 39539 → 80 [SYN, ECN, CWR, RST] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39539 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Previous segment not captured] 39541 → 80 [<None>] Seq=2295984856 Win=131072 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39542 → 80 [FIN, SYN, PSH, URG] Seq=2295984856 Win=131072 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Previous segment not captured] 39543 → 80 [ACK] Seq=2295984856 Ack=4154496 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39544 → 1 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Previous segment not captured] 39545 → 1 [ACK] Seq=2295984856 Ack=4154496 Len=0 |

| Destination | Protocol | Length | Info |
|--------------|----------|--------|---|
| 203.0.113.20 | TCP | 74 | [TCP Retransmission] 39542 → 80 [FIN, SYN, PSH, URG] Seq=2295984856 Win=131072 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39527 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39527 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39528 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39528 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39529 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39529 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 70 | [TCP Port numbers reused] 39530 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39530 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39531 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39531 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 70 | [TCP Port numbers reused] 39532 → 80 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39532 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | ICMP | 162 | Echo (ping) request id=0x08c8, seq=295/9985, ttl=58 (|
| 203.0.113.20 | ICMP | 192 | Echo (ping) request id=0x08c9, seq=296/10241, ttl=58 (|
| 203.0.113.20 | UDP | 342 | 39465 → 38585 Len=300 |
| 203.0.113.20 | TCP | 66 | [TCP Port numbers reused] 39539 → 80 [SYN, ECN, CWR, RST] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 54 | 39539 → 80 [RST] Seq=1 Win=0 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Previous segment not captured] 39541 → 80 [<None>] Seq=2295984856 Win=131072 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39542 → 80 [FIN, SYN, PSH, URG] Seq=2295984856 Win=131072 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Previous segment not captured] 39543 → 80 [ACK] Seq=2295984856 Ack=4154496 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Port numbers reused] 39544 → 1 [SYN] Seq=0 Win=0 Len=0 |
| 203.0.113.20 | TCP | 74 | [TCP Previous segment not captured] 39545 → 1 [ACK] Seq=2295984856 Ack=4154496 Len=0 |

Part 4: Perform Nmap OS and Service Foot Printing-Cont



d. We can sometimes identify a device by looking at the time to live (TTL) field of a local ping response, which varies by device OS. See the table below:

| Operating System | TTL Response time |
|------------------|-------------------|
| Cisco | 255 |
| Windows | 128 |
| Linux | 64 |

| ip.src == 203.0.113.1 && ip.dst == 203.0.113.20 | | | | | | | |
|---|--------------|-------------|--------------|----------|--------|---------------------|--|
| No. | Time | Source | Destination | Protocol | Length | Info | |
| 2370 | 57.766444894 | 203.0.113.1 | 203.0.113.20 | TCP | 74 | [TCP Retransmission | |
| 2375 | 58.900733991 | 203.0.113.1 | 203.0.113.20 | TCP | 74 | [TCP Port numbers r | |
| 2377 | 58.902272146 | 203.0.113.1 | 203.0.113.20 | TCP | 54 | 39527 → 80 [RST] Se | |
| 2378 | 59.001224563 | 203.0.113.1 | 203.0.113.20 | TCP | 74 | [TCP Port numbers r | |
| 2380 | 59.002515954 | 203.0.113.1 | 203.0.113.20 | TCP | 54 | 39528 → 80 [RST] Se | |
| 2381 | 59.101802007 | 203.0.113.1 | 203.0.113.20 | TCP | 74 | [TCP Port numbers r | |
| 2383 | 59.103126969 | 203.0.113.1 | 203.0.113.20 | TCP | 54 | 39529 → 80 [RST] Se | |
| 2384 | 59.202013257 | 203.0.113.1 | 203.0.113.20 | TCP | 70 | [TCP Port numbers r | |
| 2386 | 59.203346704 | 203.0.113.1 | 203.0.113.20 | TCP | 54 | 39530 → 80 [RST] Se | |
| 2387 | 59.302705819 | 203.0.113.1 | 203.0.113.20 | TCP | 74 | [TCP Port numbers r | |
| 2389 | 59.304401086 | 203.0.113.1 | 203.0.113.20 | TCP | 54 | 39531 → 80 [RST] Se | |
| 2390 | 59.403054020 | 203.0.113.1 | 203.0.113.20 | TCP | 70 | [TCP Port numbers r | |
| 2392 | 59.404461953 | 203.0.113.1 | 203.0.113.20 | TCP | 54 | 39532 → 80 [RST] Se | |
| 2393 | 59.428584161 | 203.0.113.1 | 203.0.113.20 | ICMP | 162 | Echo (ping) request | |
| 2395 | 59.454539975 | 203.0.113.1 | 203.0.113.20 | ICMP | 192 | Echo (ping) request | |
| 2397 | 59.480052309 | 203.0.113.1 | 203.0.113.20 | UDP | 342 | 39465 → 38585 Len=3 | |
| 2399 | 59.505538246 | 203.0.113.1 | 203.0.113.20 | TCP | 66 | [TCP Port numbers r | |
| 2401 | 59.507043776 | 203.0.113.1 | 203.0.113.20 | TCP | 54 | 39539 → 80 [RST] Se | |
| 2402 | 59.531251497 | 203.0.113.1 | 203.0.113.20 | TCP | 74 | [TCP Previous segme | |
| 2403 | 59.556932552 | 203.0.113.1 | 203.0.113.20 | TCP | 74 | [TCP Port numbers r | |
| 2404 | 59.582093406 | 203.0.113.1 | 203.0.113.20 | TCP | 74 | [TCP Previous segme | |
| 2406 | 59.607698522 | 203.0.113.1 | 203.0.113.20 | TCP | 74 | [TCP Port numbers r | |
| 2408 | 59.633599379 | 203.0.113.1 | 203.0.113.20 | TCP | 74 | [TCP Previous segme | |

Part 4: Perform Nmap OS and Service Foot Printing-cont



In the Kali VM terminal, enter the command to ping your Raspberry Pi with 4 ICMP packets.

root@kali:~# `ping -c4 203.0.113.20`

```
root@kali:~# ping -c4 203.0.113.20
PING 203.0.113.20 (203.0.113.20) 56(84) bytes of data:
64 bytes from 203.0.113.20: icmp_seq=1 ttl=64 time=1.15 ms
64 bytes from 203.0.113.20: icmp_seq=2 ttl=64 time=1.78 ms
64 bytes from 203.0.113.20: icmp_seq=3 ttl=64 time=1.31 ms
64 bytes from 203.0.113.20: icmp_seq=4 ttl=64 time=0.859 ms

--- 203.0.113.20 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 0.859/1.279/1.787/0.337 ms
```

What is the response time? Is it consistent with the Nmap identification?

The response time is 64 which is consistent with a Linux OS response.

e. Stop the Wireshark capture when finished.

Part 4: Perform Nmap OS and Service Foot Printing-Cont



Step 2: Use Nmap to find services versions

Nmap has a built-in database of about 2,200 well-known services that is used to help identify application service versions.

- Start a new Wireshark capture. Click Continue without Saving when prompted to save the capture. Apply the same display filter as the previous step.
- In the Kali VM, enter the nmap command with the -A option scanning the IP address of your Raspberry Pi.

```
root@kali:~# nmap -A 203.0.113.20
```

```
root@kali:~# nmap -A 203.0.113.20
Starting Nmap 7.60 ( https://nmap.org ) at 2024-03-12 21:15 EDT
Nmap scan report for 203.0.113.20
Host is up (0.0015s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      Tornado httpd 6.0.3
| http-robots.txt: 1 disallowed entry
| /
```

Part 4: Perform Nmap OS and Service Foot Printing-cont



What are the identified applications running on the ports? Complete the table.

| Port Number | Application service identified |
|-------------|--------------------------------|
| 22 | SSH-2.0-OpenSSH_7.4p1 |
| 80 | Tornado httpd 4.5.1 |
| 88 | nginx 1.10.3 |
| 89 | Node.js Express framework |

Answers may vary, but ports 22 and 80 should at least be open.

```
root@kali:~# nmap -A 203.0.113.20
Starting Nmap 7.60 ( https://nmap.org ) at 2024-03-12 21:15 EDT
Nmap scan report for 203.0.113.20
Host is up (0.0015s latency):
Not shown: 999 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http    Tornado httpd 6.0.3
| http-robots.txt: 1 disallowed entry
|_/
| http-server-header: TornadoServer/6.0.3
| http-title: Jupyter Notebook
|_Requested resource was /login?next=%2Ftree%3F
MAC Address: B8:27:EB:6D:7A:55 (Raspberry Pi Foundation)
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/
TCP/IP fingerprint:
OS:SCAN(V=7.60%E=4%D=3/12%OT=80%CT=1%CU=43066%PV=N%DS=1%DC=D%G=Y%M=B827EB%T
OS:M=65F0FE47%P=x86_64-pc-linux-gnu)SEQ(SP=107%GCD=1%ISR=10C%TI=Z%CI=Z%II=I
OS:%TS=A)OPS(O1=M5B4ST11NW7%O2=M5B4ST11NW7%O3=M5B4NNT11NW7%O4=M5B4ST11NW7%O
OS:5=M5B4ST11NW7%O6=M5B4ST11)WIN(W1=FE88%W2=FE88%W3=FE88%W4=FE88%W5=FE88%W6
OS:=FE88)ECN(R=Y%DF=Y%T=40%W=FAF0%O=M5B4NNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=0
OS:%A=S+F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=
OS:0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%
OS:S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+F=AR%O=%RD=0%Q=)U1(
OS:R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=
OS:N%T=40%CD=S)

Network Distance: 1 hop

TRACEROUTE
HOP RTT ADDRESS
1 1.54 ms 203.0.113.20

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 32.11 seconds
```


Part 4: Perform Nmap OS and Service Foot Printing



What is different in the port identification from the Nmap TCP scan in Part 2 of the lab and this application service scan? What implications does this have for IoT security?

In the normal TCP scan, only protocols are identified. In this scan, details about the running services are also shown. These details can be used to identify versions of software services that have well-known vulnerabilities.

What additional functions does this scan also perform?

Nmap also performs a traceroute, OS detection, and SSH details.

| | Time | Source | Destination | Protocol | Length | Info |
|------|---------------|--------------|--------------|--------------|--------------|---|
| 8835 | 213.198317868 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 54138 → 3351 [SYN] Seq=0 |
| 8836 | 213.198462745 | 203.0.113.1 | 203.0.113.20 | TCP | 58 | 54138 → 1805 [SYN] Seq=0 |
| 8837 | 213.198592697 | 203.0.113.20 | 203.0.113.1 | TCP | 60 | 1093 → 54138 [RST, ACK] |
| 8838 | 213.198596354 | 203.0.113.20 | 203.0.113.1 | TCP | 60 | 8600 → 54138 [RST, ACK] |
| 8839 | 213.198658131 | 203.0.113.1 | | | | |
| 8840 | 213.198798868 | 203.0.113.1 | 491694292 | 203.0.113.1 | 203.0.113.20 | TCP 74 [TCP Port numbers reused] 36951 → |
| 8841 | 213.199036833 | 203.0.113.20 | 493327608 | 203.0.113.20 | 203.0.113.1 | TCP 74 80 → 36951 [SYN, ACK] Seq=0 Ack=1 |
| 8842 | 213.199039155 | 203.0.113.20 | 493354777 | 203.0.113.1 | 203.0.113.20 | TCP 54 36951 → 80 [RST] Seq=1 Win=0 Len=0 |
| 8843 | 213.199039889 | 203.0.113.20 | 527777912 | 203.0.113.19 | 224.0.0.251 | MDNS 80 Standard query 0x0000 A arifRP11. |
| 8844 | 213.199101400 | 203.0.113.1 | 596127488 | 203.0.113.1 | 203.0.113.20 | TCP 74 [TCP Port numbers reused] 36952 → |
| 8845 | 213.199238389 | 203.0.113.1 | 597157836 | 203.0.113.20 | 203.0.113.1 | TCP 74 80 → 36952 [SYN, ACK] Seq=0 Ack=1 |
| 8846 | 213.199381729 | 203.0.113.1 | 597181143 | 203.0.113.1 | 203.0.113.20 | TCP 54 36952 → 80 [RST] Seq=1 Win=0 Len=0 |
| 8847 | 213.199485204 | 203.0.113.20 | 598523573 | 203.0.113.19 | 20.189.173.2 | TCP 66 57216 → 443 [SYN] Seq=0 Win=64240 |
| 8848 | 213.199487616 | 203.0.113.20 | 598731481 | 203.0.113.19 | 8.8.8.8 | DNS 91 Standard query 0xa7f7 AAAA teams. |
| 8849 | 213.199488431 | 203.0.113.20 | 596625539 | 203.0.113.1 | 203.0.113.20 | TCP 74 [TCP Port numbers reused] 36953 → |
| 8850 | 213.199547265 | 203.0.113.1 | 598279808 | 203.0.113.20 | 203.0.113.1 | TCP 74 80 → 36953 [SYN, ACK] Seq=0 Ack=1 |
| 8851 | 213.199684284 | 203.0.113.1 | 598312774 | 203.0.113.1 | 203.0.113.20 | TCP 54 36953 → 80 [RST] Seq=1 Win=0 Len=0 |
| 8852 | 213.199840838 | 203.0.113.1 | 797259177 | 203.0.113.1 | 203.0.113.20 | TCP 70 [TCP Port numbers reused] 36954 → |
| 8853 | 213.199944988 | 203.0.113.20 | 798890979 | 203.0.113.20 | 203.0.113.1 | TCP 74 80 → 36954 [SYN, ACK] Seq=0 Ack=1 |
| 8854 | 213.199947705 | 203.0.113.20 | 798922745 | 203.0.113.1 | 203.0.113.20 | TCP 54 36954 → 80 [RST] Seq=1 Win=0 Len=0 |
| 8855 | 213.199949049 | 203.0.113.20 | 898388265 | 203.0.113.1 | 203.0.113.20 | TCP 74 [TCP Port numbers reused] 36955 → |
| 8856 | 213.200007783 | 203.0.113.1 | 900032986 | 203.0.113.20 | 203.0.113.1 | TCP 74 80 → 36955 [SYN, ACK] Seq=0 Ack=1 |
| 8857 | 213.200227273 | 203.0.113.1 | 900066332 | 203.0.113.1 | 203.0.113.20 | TCP 54 36955 → 80 [RST] Seq=1 Win=0 Len=0 |
| 8858 | 213.200388718 | 203.0.113.20 | 998860401 | 203.0.113.1 | 203.0.113.20 | TCP 70 [TCP Port numbers reused] 36956 → |
| 8859 | 213.200488718 | 203.0.113.20 | 999655087 | 203.0.113.20 | 203.0.113.1 | TCP 70 80 → 36956 [SYN, ACK] Seq=0 Ack=1 |
| 8860 | 213.200488718 | 203.0.113.20 | 999672788 | 203.0.113.1 | 203.0.113.20 | TCP 54 36956 → 80 [RST] Seq=1 Win=0 Len=0 |
| 8861 | 213.200488718 | 203.0.113.20 | 924640944 | 203.0.113.1 | 203.0.113.20 | ICMP 162 Echo (ping) request id=0xcba3, s |
| 8862 | 213.200488718 | 203.0.113.20 | 926049039 | 203.0.113.20 | 203.0.113.1 | ICMP 162 Echo (ping) reply id=0xcba3, s |
| 8863 | 213.200488718 | 203.0.113.20 | 958181858 | 203.0.113.1 | 203.0.113.20 | TCP 100 54138 → 3351 [SYN] Seq=0 |

c. Stop the Wireshark capture when finished.

References



1. What is a computer port? | Ports in networking: <https://www.cloudflare.com/learning/network-layer/what-is-a-computer-port/>
2. Oracle Agile Engineering Data Management Security Guide:
https://docs.oracle.com/cd/E89228_03/otn/pdf/install/html_edmsc/output/chapter_6.htm
3. Computer Ports and Cables
<https://elluminetpress.com/2017/08/computer-ports-and-cables/>
4. Port Scanning an IoT Device
<https://www.youtube.com/watch?v=1XisyWvY0SY>

