Nmap Live Host Discovery

Learn how to use Nmap to discover live hosts using ARP scan, ICMP scan, and TCP/UDP ping scan.

Task 1 Introduction

When we want to target a network, we want to find an efficient tool to help us handle repetitive tasks and answer the following questions:

1. Which systems are up?
2. What services are running on these systems?

The tool that we will rely on is Nmap. The first question about finding live computers is answered in this room. This room is the first in a series of four rooms dedicated to Nmap. The second question about discovering running services is answered in the next Nmap rooms that focus on port-scanning.

This room is the first of four in this Nmap series. These four rooms are also part of the Network Security module.

1. Nmap Live Host Discovery
2. Nmap Basic Port Scans
3. Nmap Advanced Port Scans
4. Nmap Post Port Scans

This room explains the steps that Nmap carries out to discover the systems that are online before port-scanning. This stage is crucial because trying to port-scan offline systems will only waste time and create unnecessary noise on the network.
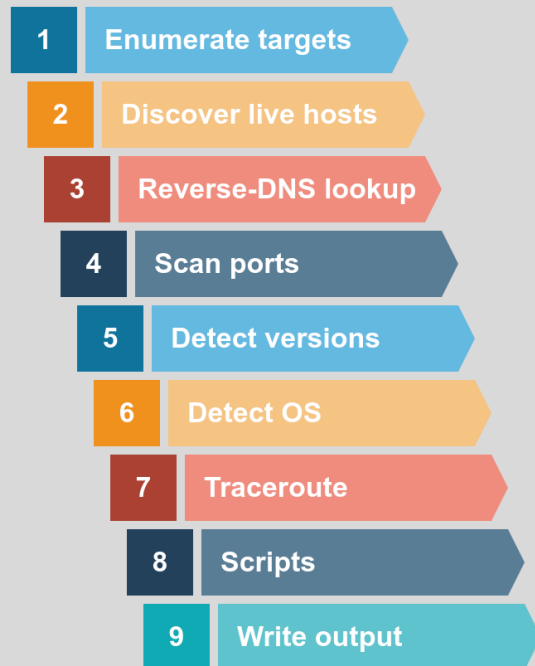
We present the different approaches that Nmap uses to discover live hosts. In particular, we cover:

1. ARP scan: This scan uses ARP requests to discover live hosts
2. ICMP scan: This scan uses ICMP requests to identify live hosts
3. TCP/UDP ping scan: This scan sends packets to TCP ports and UDP ports to determine live hosts.

We also introduce two scanners, arp-scan and masscan, and explain how they overlap with part of Nmap's host discovery.

As already mentioned, starting with this room, we will use Nmap to discover systems and services actively. Nmap was created by Gordon Lyon (Fyodor), a network security expert and open source programmer. It was released in 1997. Nmap, short for Network Mapper, is free, open-source software released under GPL license. Nmap is an industry-standard tool for mapping networks, identifying live hosts, and discovering running services. Nmap's scripting engine can further extend its functionality, from

fingerprinting services to exploiting vulnerabilities. A Nmap scan usually goes through the steps shown in the figure below, although many are optional and depend on the command-line arguments you provide.

1	Enumerate targets
2	Discover live hosts
3	Reverse-DNS lookup
4	Scan ports
5	Detect versions
6	Detect OS
7	Traceroute
8	Scripts
9	Write output

Answer the questions below

Some of these questions will require the use of a static site to answer the task questions, while others require the use of the AttackBox.

No answer needed

Task 2 Subnetworks

Let's review a couple of terms before we move on to the main tasks. A network segment is a group of computers connected using a shared medium. For instance, the medium can be the Ethernet switch or WiFi access point. In an IP network, a subnetwork is usually the equivalent of one or more network segments connected together and configured to use the same router. The network segment refers to a physical connection, while a subnetwork refers to a logical connection.

In the following network diagram, we have four network segments or subnetworks. Generally speaking, your system would be connected to one of these network segments/subnetworks. A subnetwork, or simply a subnet, has its own IP address range and is connected to a more extensive network via a router. There might be a firewall enforcing security policies depending on each network.

The figure above shows two types of subnets:

- Subnets with /16, which means that the subnet mask can be written as 255.255.0.0. This subnet can have around 65 thousand hosts.
- Subnets with /24, which indicates that the subnet mask can be expressed as 255.255.255.0. This subnet can have around 250 hosts.
-

You might want to refer to Task 2 in the Intro to LAN room if you need to learn more about subnetting.

As part of active reconnaissance, we want to discover more information about a group of hosts or about a subnet. If you are connected to the same subnet, you would expect your scanner to rely on ARP (Address Resolution Protocol) queries to discover live hosts. An ARP query aims to get the hardware address (MAC address) so that communication over the link-layer becomes possible; however, we can use this to infer that the host is online. (We revisit link-layer in Task 4.)

If you are in Network A, you can use ARP only to discover the devices within that subnet (10.1.100.0/24). Suppose you are connected to a subnet different from the subnet of the target system(s). In that case, all packets generated by your scanner will be routed via the default gateway (router) to reach the systems on another subnet; however, the ARP queries won't be routed and hence cannot cross the subnet router. ARP is a link-layer protocol, and ARP packets are bound to their subnet.

Click on the "View Site" button to start the network simulator. We will use this simulator to answer the questions in tasks 2, 4, and 5.

Answer the questions below

Send a packet with the following:

Send Packet

From:
computer1

To:
computer1

Packet Type:
arp_request

Data:
computer6

Send Packet

- From computer1
- To computer1 (to indicate it is broadcast)
- Packet Type: "ARP Request"
- Data: computer6 (because we are asking for computer6 MAC address using ARP Request)
- How many devices can see the ARP Request?

How many devices can see the ARP Request?

4

Send a packet with the following:

- From computer4
- To computer4 (to indicate it is broadcast)
- Packet Type: "ARP Request"
- Data: computer6 (because we are asking for computer6 MAC address using ARP Request)

How many devices can see the ARP Request?

4

Task 3 Enumerating Targets

We mentioned the different techniques we can use for scanning in Task 1. Before we explain each in detail and put it into use against a live target, we need to specify the targets we want to scan. Generally speaking, you can provide a list, a range, or a subnet. Examples of target specification are:

- **list**: **nmap MACHINE_IP scanme.nmap.org example.com** will scan 3 IP addresses.

- **range: nmap 10.11.12.15-20** will scan 6 IP addresses: 10.11.12.15, 10.11.12.16,... and 10.11.12.20.

- **subnet**: **nmap MACHINE_IP/30** will scan 4 IP addresses.

You can also provide a **file as input** for your list of targets, **nmap -iL list_of_hosts.txt.**

If you want to **check the list of hosts** that Nmap will scan, you can use **nmap -sL TARGETS**. This option will give you a detailed list of the hosts that Nmap will scan without scanning them; however, Nmap will attempt a reverse-DNS resolution on all the targets to obtain their names. Names might reveal various information to the pentester. (If you don't want Nmap to the DNS server, you can add -n.)

Answer the questions below

What is the first IP address Nmap would scan if you provided `10.10.12.13/29` as your target?
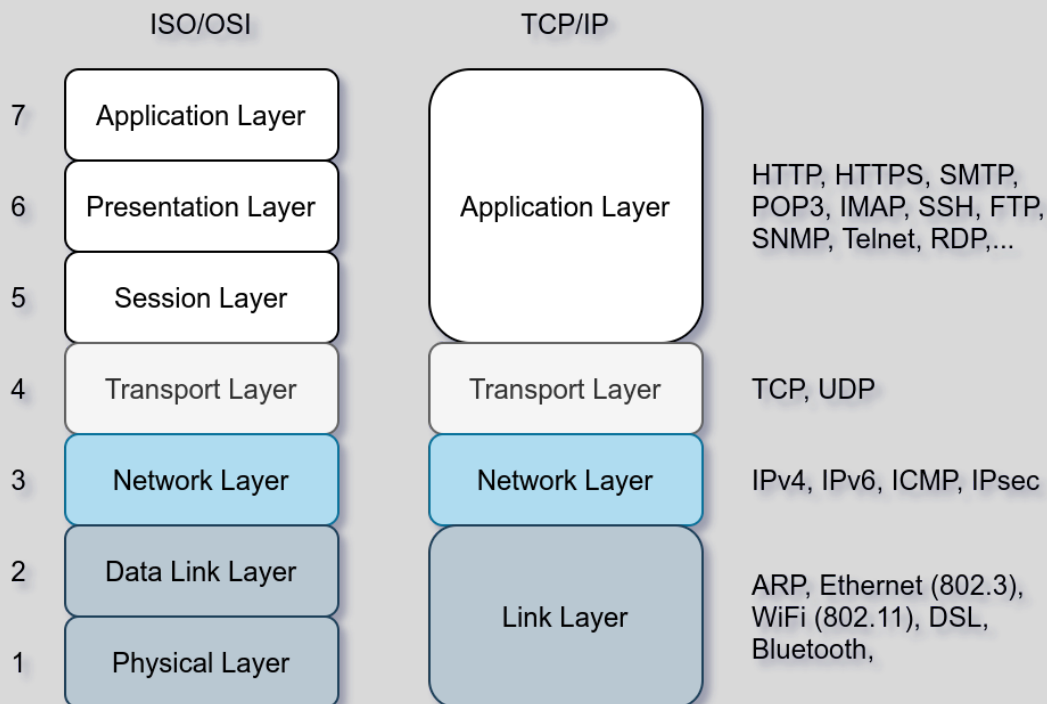
10.10.12.8

How many IP addresses will Nmap scan if you provide the following range 10.10.0-255.101-125?

6400

Task 4 Discovering Live Hosts

Let's revisit the TCP/IP layers shown in the figure next. We will leverage the protocols to discover the live hosts. Starting from bottom to top, we can use:

- ARP from Link Layer
- ICMP from Network Layer
- TCP from Transport Layer
- UDP from Transport Layer

	ISO/OSI	TCP/IP	
7	Application Layer	Application Layer	HTTP, HTTPS, SMTP, POP3, IMAP, SSH, FTP, SNMP, Telnet, RDP,...
6	Presentation Layer		
5	Session Layer		
4	Transport Layer	Transport Layer	TCP, UDP
3	Network Layer	Network Layer	IPv4, IPv6, ICMP, IPsec
2	Data Link Layer	Link Layer	ARP, Ethernet (802.3), WiFi (802.11), DSL, Bluetooth,
1	Physical Layer		

Before we discuss how scanners can use each in detail, we will briefly review these four protocols. ARP has one purpose: sending a frame to the broadcast address on the network segment and asking the computer with a specific IP address to respond by providing its MAC (hardware) address.

ICMP has many types. ICMP ping uses Type 8 (Echo) and Type 0 (Echo Reply).

If you want to ping a system on the same subnet, an ARP query should precede the ICMP Echo.

Although TCP and UDP are transport layers, for network scanning purposes, a scanner can send a specially-crafted packet to common TCP or UDP ports to check whether the target will respond. This method is efficient, especially when ICMP Echo is blocked.

If you have closed the network simulator, click on the "View Site" button in Task 2 to display it again.

Answer the questions below

Send a packet with the following:

- From computer1
- To computer3
- Packet Type: "Ping Request"

What is the type of packet that computer1 sent before the ping?

ARP Request

What is the type of packet that computer1 received before being able to send the ping?

ARP Request

How many computers responded to the ping request?

1

Send a packet with the following:

- From computer2
- To computer5
- Packet Type: "Ping Request"

What is the name of the first device that responded to the first ARP Request?

router

What is the name of the first device that responded to the second ARP Request?

computer5

Send another Ping Request. Did it require new ARP Requests? (Y/N)

N

Task 5 Nmap Host Discovery Using ARP

How would you know which hosts are up and running? It is essential to avoid wasting our time port-scanning an offline host or an IP address not in use. There are various ways to discover online hosts. When no host discovery options are provided, Nmap follows the following approaches to discover live hosts:

1. When a privileged user tries to scan targets on a local network (Ethernet), Nmap uses ARP requests. A privileged user is root or a user who belongs to sudoers and can run sudo.
2. When a privileged user tries to scan targets outside the local network, Nmap uses ICMP echo requests, TCP ACK (Acknowledge) to port 80, TCP SYN (Synchronize) to port 443, and ICMP timestamp request.
3. When an unprivileged user tries to scan targets outside the local network, Nmap resorts to a TCP 3-way handshake by sending SYN packets to ports 80 and 443.

Nmap, by default, uses a ping scan to find live hosts, then proceeds to scan live hosts only. If you want to use Nmap to discover online hosts without port-scanning the live systems, you can issue nmap -sn TARGETS. Let's dig deeper to gain a solid understanding of the different techniques used.

ARP scan is possible only if you are on the same subnet as the target systems. On an Ethernet (802.3) and WiFi (802.11), you need to know the MAC address of any system before you can communicate with it. The MAC address is necessary for the link-layer header; the header contains the source MAC address and the destination MAC address among other fields. To get the MAC address, the OS sends an ARP query. A host that replies to ARP queries is up. The ARP query only works if the target is on the same subnet as yourself, i.e., on the same Ethernet/WiFi. You should expect to see many ARP queries generated during a Nmap scan of a local network. If you want Nmap only to perform an A**RP scan without port-scanning**, you can use **nmap -PR -sn TARGETS**, where -PR indicates that you only want an ARP scan. The following example shows Nmap using ARP for host discovery without any port scanning. We run nmap -PR -sn MACHINE_IP/24 to discover all the live systems on the same subnet as our target machine.

```
pentester@TryHackMe$ sudo nmap -PR -sn 10.10.210.6/24

Starting Nmap 7.60 ( https://nmap.org ) at 2021-09-02 07:12 BST
Nmap scan report for ip-10-10-210-75.eu-west-1.compute.internal (10.10.210.75)
Host is up (0.00013s latency).
MAC Address: 02:83:75:3A:F2:89 (Unknown)
Nmap scan report for ip-10-10-210-100.eu-west-1.compute.internal
(10.10.210.100)
Host is up (-0.100s latency).
MAC Address: 02:63:D0:1B:2D:CD (Unknown)
Nmap scan report for ip-10-10-210-165.eu-west-1.compute.internal
(10.10.210.165)
Host is up (0.00025s latency).
MAC Address: 02:59:79:4F:17:B7 (Unknown)
Nmap scan report for ip-10-10-210-6.eu-west-1.compute.internal (10.10.210.6)
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 3.12 seconds
```

In this case, the AttackBox had the IP address 10.10.210.6, and it used ARP requests to discover the live hosts on the same subnet. ARP scan works, as shown in the figure below. Nmap sends ARP requests to all the target computers, and those online should send an ARP reply back.

```
nmap -PR -sn TARGET
```


ARP Request

ARP Reply

Case: Host is live.
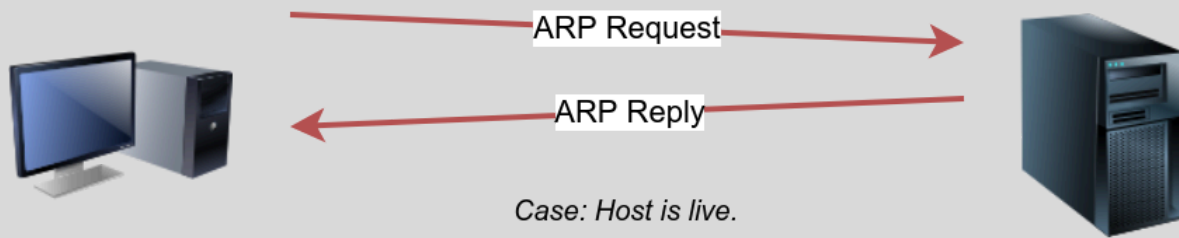
If we look at the packets generated using a tool such as tcpdump or Wireshark, we will see network traffic similar to the figure below. In the figure below, Wireshark displays the source MAC address, destination MAC address, protocol, and query related to each ARP request. The source address is the MAC address of our AttackBox, while the destination is the broadcast address as we don't know the MAC address of the target. However, we see the target's IP address, which appears in the Info column. In the figure, we can see that we are requesting the MAC addresses of all the IP addresses on the subnet, starting with 10.10.210.1. The host with the IP address we are asking about will send an ARP reply with its MAC address, and that's how we will know that it is online.

Talking about ARP scans, we should mention a scanner built around ARP queries: arp-scan; it provides many options to customize your scan. Visit the arp-scan wiki for detailed information. One popular choice

is arp-scan --localnet or simply arp-scan -l. This command will send ARP queries to all valid IP addresses on your local networks. Moreover, if your system has more than one interface and you are interested in discovering the live hosts on one of them, you can specify the interface using -I. For instance, sudo arp-scan -I eth0 -l will send ARP queries for all valid IP addresses on the eth0 interface.

Note that arp-scan is not installed on the AttackBox; however, it can be installed using apt install arp-scan.

In the example below, we scanned the subnet of the AttackBox using arp-scan ATTACKBOX_IP/24. Since we ran this scan at a time frame close to the previous one **nmap -PR -sn ATTACKBOX_IP/24**, we obtained the same three live targets.

```
pentester@TryHackMe$ sudo arp-scan 10.10.210.6/24
Interface: eth0, datalink type: EN10MB (Ethernet)
WARNING: host part of 10.10.210.6/24 is non-zero
Starting arp-scan 1.9 with 256 hosts
(http://www.nta-monitor.com/tools/arp-scan/)
10.10.210.75  02:83:75:3a:f2:89  (Unknown)
10.10.210.100    02:63:d0:1b:2d:cd  (Unknown)
10.10.210.165    02:59:79:4f:17:b7  (Unknown)

4 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9: 256 hosts scanned in 2.726 seconds (93.91 hosts/sec). 3
responded
```

Similarly, the command arp-scan will generate many ARP queries that we can see using tcpdump, Wireshark, or a similar tool. We can notice that the packet capture for arp-scan and nmap -PR -sn yield similar traffic patterns. Below is the Wireshark output.

```
arp-scan-AttackBox.pcapng
File   Edit   View   Go   Capture   Analyze   Statistics   Telephony   Wireless   Tools   Help

arp

Source                Destination     Protocol   Info
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.0? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.1? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.2? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.3? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.4? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.5? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        ARP Announcement for 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.7? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.8? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.9? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.10? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.11? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.12? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.13? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.14? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.15? Tell 10.10.210.6
02:ba:eb:d6:18:2b     Broadcast       ARP        Who has 10.10.210.16? Tell 10.10.210.6

  Address Resolution Protocol: Protocol          Packets: 1207 · Displayed: 512 (42.4%)   Profile: Default
```

If you have closed the network simulator, click on the "Visit Site" button in Task 2 to display it again.

Answer the questions below

We will be sending broadcast ARP Requests packets with the following options:

- From computer1
- To computer1 (to indicate it is broadcast)
- Packet Type: "ARP Request"
- Data: try all the possible eight devices (other than computer1) in the network: computer2, computer3, computer4, computer5, computer6, switch1, switch2, and router.

How many devices are you able to discover using ARP requests?

3

Task 6 Nmap Host Discovery Using ICMP

We can ping every IP address on a target network and see who would respond to our ping (ICMP Type 8/Echo) requests with a ping reply (ICMP Type 0). Simple, isn't it? Although this would be the most straightforward approach, it is not always reliable. Many firewalls block ICMP echo; new versions of MS Windows are configured with a host firewall that blocks ICMP echo requests by default. Remember that an ARP query will precede the ICMP request if your target is on the same subnet.

To use ICMP echo request to discover live hosts, add the option -PE. (Remember to add -sn if you don't want to follow that with a port scan.) As shown in the following figure, an ICMP echo scan works by sending an ICMP echo request and expects the target to reply with an ICMP echo reply if it is online.

nmap -PE -sn TARGET

ICMP Echo Request

ICMP Echo Reply

Case: Host is live.
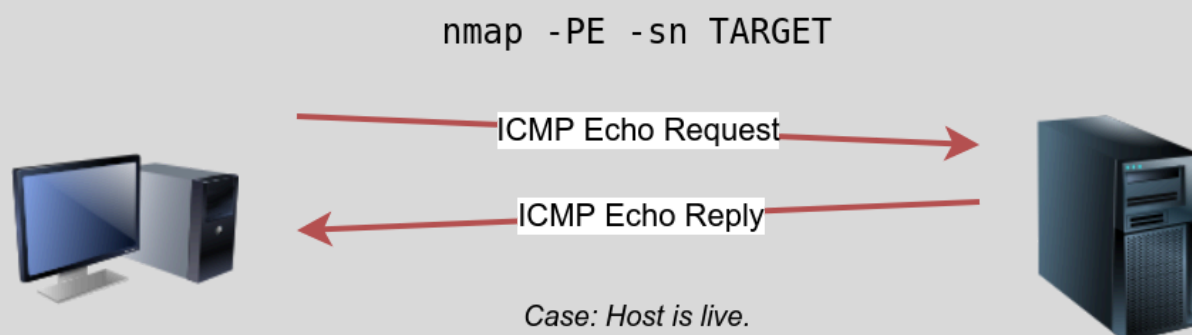
In the example below, we scanned the **target's subnet** using **nmap -PE -sn MACHINE_IP/24**. This scan will send ICMP echo packets to every IP address on the subnet. Again, we expect live hosts to reply; however, it is wise to remember that many firewalls block ICMP. The output below shows the result of scanning the virtual machine's class C subnet using **sudo nmap -PE -sn MACHINE_IP/24** from the AttackBox.

```
pentester@TryHackMe$ sudo nmap -PE -sn 10.10.68.220/24

Starting Nmap 7.60 ( https://nmap.org ) at 2021-09-02 10:16 BST
Nmap scan report for ip-10-10-68-50.eu-west-1.compute.internal (10.10.68.50)
Host is up (0.00017s latency).
MAC Address: 02:95:36:71:5B:87 (Unknown)
Nmap scan report for ip-10-10-68-52.eu-west-1.compute.internal (10.10.68.52)
Host is up (0.00017s latency).
MAC Address: 02:48:E8:BF:78:E7 (Unknown)
Nmap scan report for ip-10-10-68-77.eu-west-1.compute.internal (10.10.68.77)
Host is up (-0.100s latency).
MAC Address: 02:0F:0A:1D:76:35 (Unknown)
Nmap scan report for ip-10-10-68-110.eu-west-1.compute.internal (10.10.68.110)
Host is up (-0.10s latency).
MAC Address: 02:6B:50:E9:C2:91 (Unknown)
Nmap scan report for ip-10-10-68-140.eu-west-1.compute.internal (10.10.68.140)
Host is up (0.00021s latency).
MAC Address: 02:58:59:63:0B:6B (Unknown)
Nmap scan report for ip-10-10-68-142.eu-west-1.compute.internal (10.10.68.142)
Host is up (0.00016s latency).
MAC Address: 02:C6:41:51:0A:0F (Unknown)
Nmap scan report for ip-10-10-68-220.eu-west-1.compute.internal (10.10.68.220)
Host is up (0.00026s latency).
MAC Address: 02:25:3F:DB:EE:0B (Unknown)
Nmap scan report for ip-10-10-68-222.eu-west-1.compute.internal (10.10.68.222)
Host is up (0.00025s latency).
MAC Address: 02:28:B1:2E:B0:1B (Unknown)
```

```
Nmap done: 256 IP addresses (8 hosts up) scanned in 2.11 seconds
```

The scan output shows that eight hosts are up; moreover, it shows their MAC addresses. Generally speaking, we don't expect to learn the MAC addresses of the targets unless they are on the same subnet as our system. The output above indicates that Nmap didn't need to send ICMP packets as it confirmed that these hosts are up based on the ARP responses it received.

We will repeat the scan above; however, this time, we will scan from a system that belongs to a different subnet. The results are similar but without the MAC addresses.
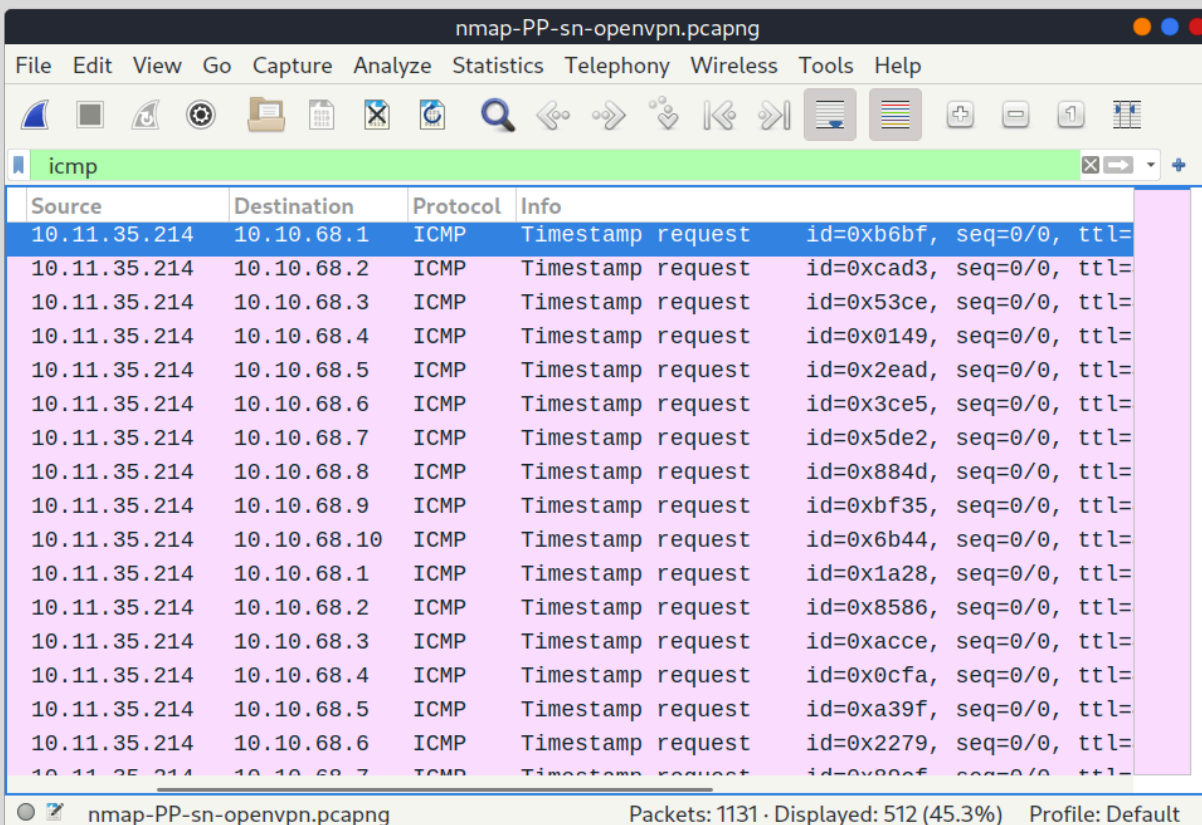
```
pentester@TryHackMe$ sudo nmap -PE -sn 10.10.68.220/24

Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-02 12:16 EEST
Nmap scan report for 10.10.68.50
Host is up (0.12s latency).
Nmap scan report for 10.10.68.52
Host is up (0.12s latency).
Nmap scan report for 10.10.68.77
Host is up (0.11s latency).
Nmap scan report for 10.10.68.110
Host is up (0.11s latency).
Nmap scan report for 10.10.68.140
Host is up (0.11s latency).
Nmap scan report for 10.10.68.142
Host is up (0.11s latency).
Nmap scan report for 10.10.68.220
Host is up (0.11s latency).
Nmap scan report for 10.10.68.222
Host is up (0.11s latency).
Nmap done: 256 IP addresses (8 hosts up) scanned in 8.26 seconds
```

If you look at the network packets using a tool like Wireshark, you will see something similar to the image below. You can see that we have one source IP address on a different subnet than that of the destination subnet, sending ICMP echo requests to all the IP addresses in the target subnet to see which one will reply.

The Wireshark window titled `nmap-PE-sn-openvpn.pcapng` showing a filtered list of ICMP packets:

Source	Destination	Protocol	Info
10.11.35.214	10.10.68.1	ICMP	Echo (ping) request id=0x22e1, seq=0/0, ttl=
10.11.35.214	10.10.68.2	ICMP	Echo (ping) request id=0xd13b, seq=0/0, ttl=
10.11.35.214	10.10.68.3	ICMP	Echo (ping) request id=0x65c8, seq=0/0, ttl=
10.11.35.214	10.10.68.4	ICMP	Echo (ping) request id=0x2b62, seq=0/0, ttl=
10.11.35.214	10.10.68.5	ICMP	Echo (ping) request id=0x8681, seq=0/0, ttl=
10.11.35.214	10.10.68.6	ICMP	Echo (ping) request id=0x6a13, seq=0/0, ttl=
10.11.35.214	10.10.68.7	ICMP	Echo (ping) request id=0x2dbc, seq=0/0, ttl=
10.11.35.214	10.10.68.8	ICMP	Echo (ping) request id=0x2029, seq=0/0, ttl=
10.11.35.214	10.10.68.9	ICMP	Echo (ping) request id=0xf800, seq=0/0, ttl=
10.11.35.214	10.10.68.10	ICMP	Echo (ping) request id=0xca62, seq=0/0, ttl=
10.11.35.214	10.10.68.1	ICMP	Echo (ping) request id=0xe95f, seq=0/0, ttl=
10.11.35.214	10.10.68.2	ICMP	Echo (ping) request id=0x896e, seq=0/0, ttl=
10.11.35.214	10.10.68.3	ICMP	Echo (ping) request id=0xdffe, seq=0/0, ttl=
10.11.35.214	10.10.68.4	ICMP	Echo (ping) request id=0xdf2c, seq=0/0, ttl=
10.11.35.214	10.10.68.5	ICMP	Echo (ping) request id=0x4602, seq=0/0, ttl=
10.11.35.214	10.10.68.6	ICMP	Echo (ping) request id=0xd84a, seq=0/0, ttl=

Packets: 1074 · Displayed: 512 (47.7%)   Profile: Default

Because ICMP echo requests tend to be blocked, you might also consider ICMP Timestamp or ICMP Address Mask requests to tell if a system is online. Nmap uses timestamp request (ICMP Type 13) and checks whether it will get a Timestamp reply (ICMP Type 14). Adding the **-PP** option tells Nmap to use ICMP timestamp requests. As shown in the figure below, you expect live hosts to reply.

`nmap -PP -sn TARGET`

ICMP Timestamp Request
ICMP Timestamp Reply

Case: Host is live.
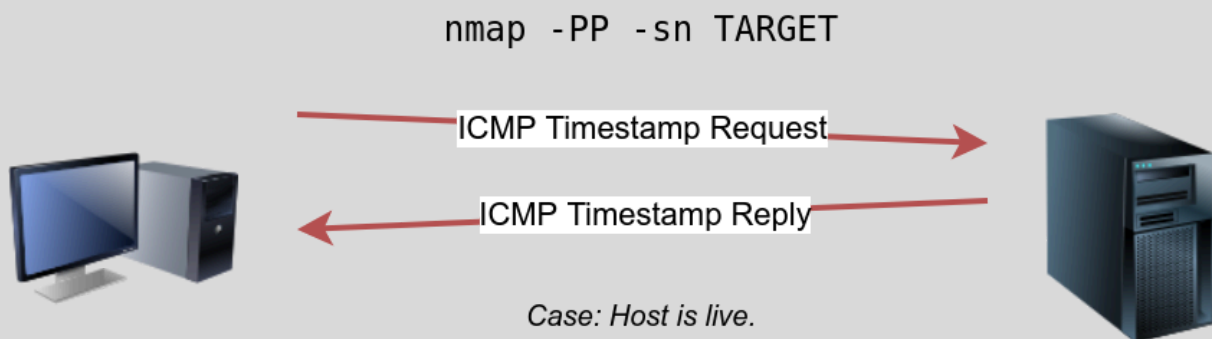
In the following example, we run **nmap -PP -sn MACHINE_IP/24** to discover the **online computers on the target machine subnet**.

```
pentester@TryHackMe$ sudo nmap -PP -sn 10.10.68.220/24

Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-02 12:06 EEST
```

```
Nmap scan report for 10.10.68.50
Host is up (0.13s latency).
Nmap scan report for 10.10.68.52
Host is up (0.25s latency).
Nmap scan report for 10.10.68.77
Host is up (0.14s latency).
Nmap scan report for 10.10.68.110
Host is up (0.14s latency).
Nmap scan report for 10.10.68.140
Host is up (0.15s latency).
Nmap scan report for 10.10.68.209
Host is up (0.14s latency).
Nmap scan report for 10.10.68.220
Host is up (0.14s latency).
Nmap scan report for 10.10.68.222
Host is up (0.14s latency).
Nmap done: 256 IP addresses (8 hosts up) scanned in 10.93 seconds
```

Similar to the previous ICMP scan, this scan will send many ICMP timestamp requests to every valid IP address in the target subnet. In the Wireshark screenshot below, you can see one source IP address sending ICMP packets to every possible IP address to discover online hosts.

Similarly, Nmap uses address mask queries (ICMP Type 17) and checks whether it gets an address mask reply (ICMP Type 18). This scan can be enabled with the option -PM. As shown in the figure below, live hosts are expected to reply to ICMP address mask requests.

nmap -PM -sn TARGET

ICMP Address Mask Request

ICMP Address Mask Reply

Case: Host is live.
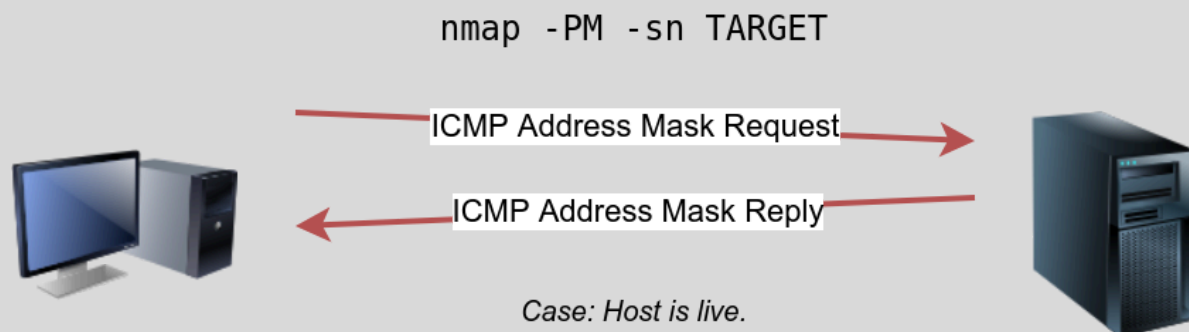
In an attempt to **discover live hosts using ICMP address mask queries**, we run the command **nmap -PM -sn MACHINE_IP/24**. Although, based on earlier scans, we know that at least eight hosts are up, this scan returned none. The reason is that the target system or a firewall on the route is blocking this type of ICMP packet. Therefore, it is essential to learn multiple approaches to achieve the same result. If one type of packet is being blocked, we can always choose another to discover the target network and services.

```
pentester@TryHackMe$ sudo nmap -PM -sn 10.10.68.220/24

Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-02 12:13 EEST
Nmap done: 256 IP addresses (0 hosts up) scanned in 52.17 seconds
```

Although we didn't get any reply and could not figure out which hosts are online, it is essential to note that this scan sent ICMP address mask requests to every valid IP address and waited for a reply. Each ICMP request was sent twice, as we can see in the screenshot below.

Answer the questions below

What is the option required to tell Nmap to use ICMP Timestamp to discover live hosts?

-PP

What is the option required to tell Nmap to use ICMP Address Mask to discover live hosts?

-PM

What is the option required to tell Nmap to use ICMP Echo to discover live hosts?

–PE

Task 7 Nmap Host Discovery Using TCP and UDP

We can send a packet with the SYN (Synchronize) flag set to a TCP port, 80 by default, and wait for a response. An open port should reply with a SYN/ACK (Acknowledge); a closed port would result in an RST (Reset). In this case, we only check whether we will get any response to infer whether the host is up. The specific state of the port is not significant here. The figure below is a reminder of how a TCP 3-way handshake usually works.

TCP 3-Way Handshake
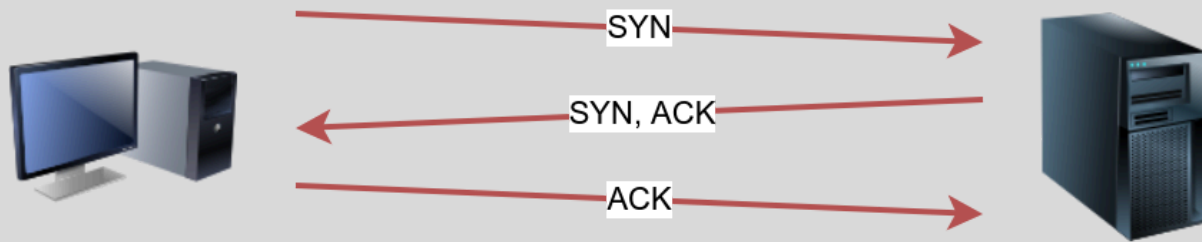
Case: TCP port is open.

If you want Nmap to use TCP SYN ping, you can do so via the option **-PS** followed by the port number, range, list, or a combination of them. For example, **-PS21 will target port 21**, while **-PS21-25 will target ports 21, 22, 23, 24, and 25**. Finally **-PS80,443,8080 will target the three ports 80, 443, and 8080**.

Privileged users (root and sudoers) can send TCP SYN packets and don't need to complete the TCP 3-way handshake even if the port is open, as shown in the figure below. Unprivileged users have no choice but to complete the 3-way handshake if the port is open.

nmap -PS -sn TARGET
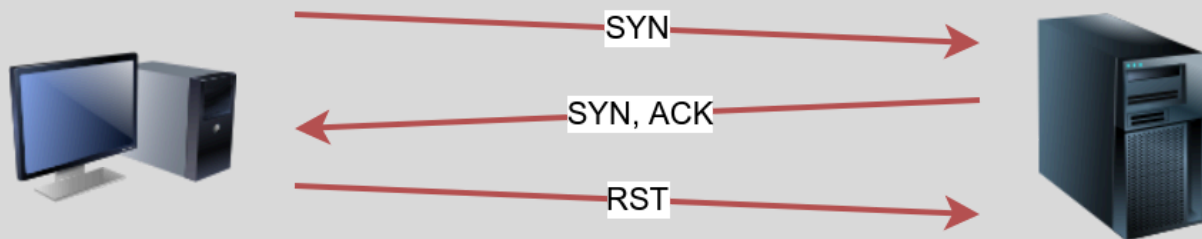
Case: TCP port is open.

We will run **nmap -PS -sn MACHINE_IP/24** to scan the **target VM subnet**. As we can see in the output below, we were able to discover five hosts.

```
pentester@TryHackMe$ sudo nmap -PS -sn 10.10.68.220/24
Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-02 13:45 EEST
Nmap scan report for 10.10.68.52
Host is up (0.10s latency).
Nmap scan report for 10.10.68.121
Host is up (0.16s latency).
Nmap scan report for 10.10.68.125
Host is up (0.089s latency).
Nmap scan report for 10.10.68.134
```

```
Host is up (0.13s latency).
Nmap scan report for 10.10.68.220
Host is up (0.11s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 17.38 seconds
```

Let's take a closer look at what happened behind the scenes by looking at the network traffic on Wireshark in the figure below. Technically speaking, since we didn't specify any TCP ports to use in the TCP ping scan, Nmap used common ports; in this case, it is TCP port 80. Any service listening on port 80 is expected to reply, indirectly indicating that the host is online.
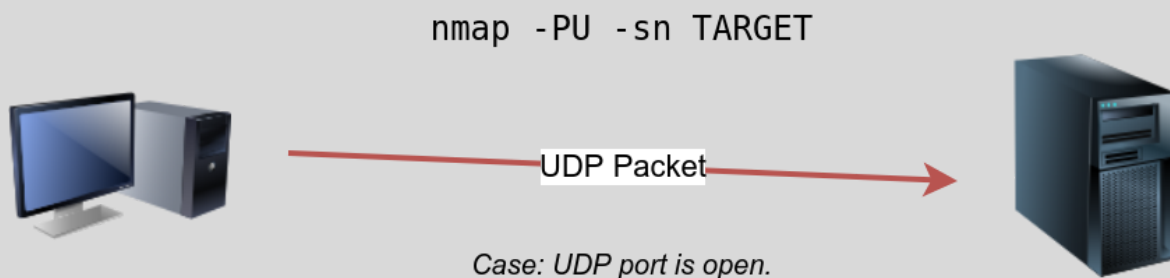

TCP ACK Ping

As you have guessed, this sends a packet with an ACK flag set. You must be running Nmap as a privileged user to be able to accomplish this. If you try it as an unprivileged user, Nmap will attempt a 3-way handshake.

By default, port 80 is used. The syntax is similar to TCP SYN ping. **-PA** should be followed by a port number, range, list, or a combination of them. For example, consider **-PA21**, **-PA21-25** and -**PA80,443,8080**. If no port is specified, port 80 will be used.

The following figure shows that any TCP packet with an ACK flag should get a TCP packet back with an RST flag set. The target responds with the RST flag set because the TCP packet with the ACK flag is not part of any ongoing connection. The expected response is used to detect if the target host is up.

```
nmap -PA -sn TARGET
```


ACK

RST

Case: TCP port is open or closed.
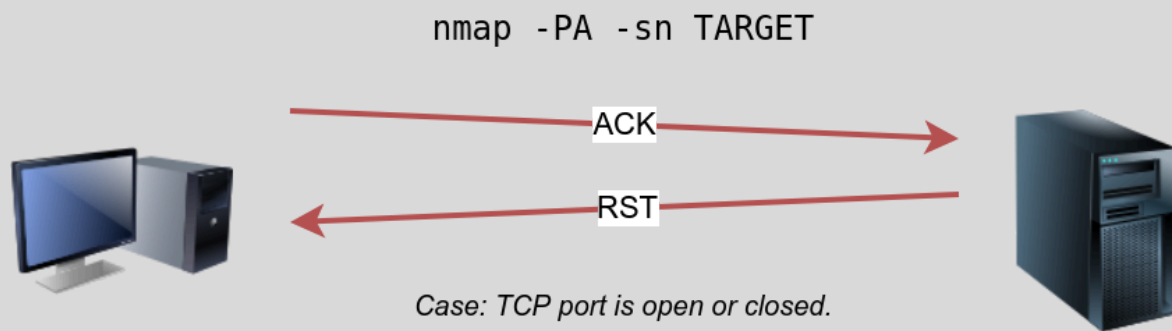
In this example, we run **sudo nmap -PA -sn MACHINE_IP/24** to discover the **online hosts on the target's subnet**. We can see that the TCP ACK ping scan detected five hosts as up.

```
pentester@TryHackMe$ sudo nmap -PA -sn 10.10.68.220/24
Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-02 13:46 EEST
Nmap scan report for 10.10.68.52
Host is up (0.11s latency).
Nmap scan report for 10.10.68.121
Host is up (0.12s latency).
Nmap scan report for 10.10.68.125
Host is up (0.10s latency).
Nmap scan report for 10.10.68.134
Host is up (0.10s latency).
Nmap scan report for 10.10.68.220
Host is up (0.10s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 29.89 seconds
```

If we peek at the network traffic as shown in the figure below, we will discover many packets with the ACK flag set and sent to port 80 of the target systems. Nmap sends each packet twice. The systems that don't respond are offline or inaccessible.
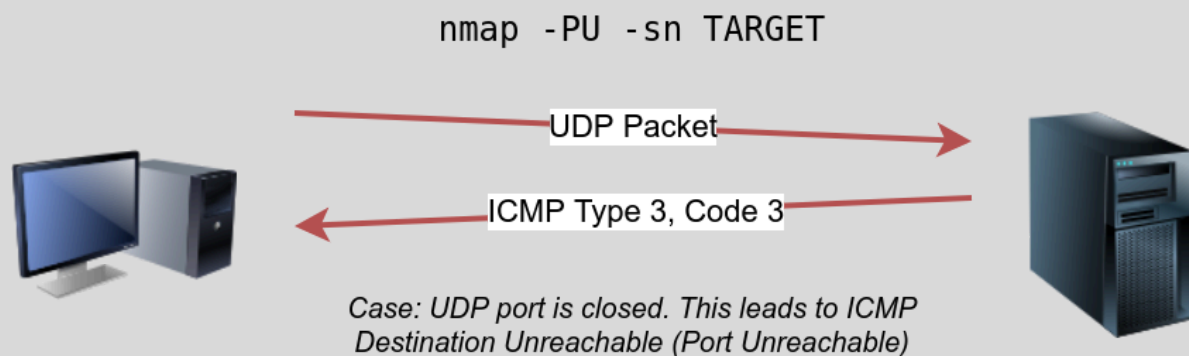
UDP Ping

Finally, we can use UDP to discover if the host is online. Contrary to TCP SYN ping, sending a UDP packet to an open port is not expected to lead to any reply. However, if we send a UDP packet to a closed UDP port, we expect to get an ICMP port unreachable packet; this indicates that the target system is up and available.

In the following figure, we see a UDP packet sent to an open UDP port and not triggering any response. However, sending a UDP packet to any closed UDP port can trigger a response indirectly indicating that the target is online.

nmap -PU -sn TARGET

UDP Packet

Case: UDP port is open.

```
nmap -PU -sn TARGET
```


UDP Packet

ICMP Type 3, Code 3

Case: UDP port is closed. This leads to ICMP
Destination Unreachable (Port Unreachable)

The syntax to specify the ports is similar to that of TCP SYN ping and TCP ACK ping; Nmap uses -PU for UDP ping. In the following example, we use a UDP scan, and we discover five live hosts.

```
pentester@TryHackMe$ sudo nmap -PU -sn 10.10.68.220/24
Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-02 13:45 EEST
Nmap scan report for 10.10.68.52
Host is up (0.10s latency).
Nmap scan report for 10.10.68.121
Host is up (0.10s latency).
Nmap scan report for 10.10.68.125
Host is up (0.14s latency).
Nmap scan report for 10.10.68.134
Host is up (0.096s latency).
Nmap scan report for 10.10.68.220
Host is up (0.11s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 9.20 seconds
```

Let's inspect the UDP packets generated. In the following Wireshark screenshot, we notice Nmap sending UDP packets to UDP ports that are most likely closed. The image below shows that Nmap uses an uncommon UDP port to trigger an ICMP destination unreachable (port unreachable) error.

Masscan

On a side note, Masscan uses a similar approach to discover the available systems. However, to finish its network scan quickly, Masscan is quite aggressive with the rate of packets it generates. The syntax is quite similar: **-p** can be followed by a port number, list, or range. Consider the following examples:

- masscan MACHINE_IP/24 -p443
- masscan MACHINE_IP/24 -p80,443
- masscan MACHINE_IP/24 -p22-25
- masscan MACHINE_IP/24 --top-ports 100

Masscan is not installed on the AttackBox; however, it can be installed using **apt install masscan**.

Answer the questions below

Which TCP ping scan does not require a privileged account?

TCP SYN Ping

Which TCP ping scan requires a privileged account?

TCP ACK Ping

What option do you need to add to Nmap to run a TCP SYN ping scan on the telnet port?

-PS23

Task 8 Using Reverse-DNS Lookup

Nmap's default behaviour is to use reverse-DNS online hosts. Because the hostnames can reveal a lot, this can be a helpful step. However, if you don't want to send such DNS queries, you use **-n** to skip this step.

By default, Nmap will look up online hosts; however, you can use the option **-R** to query the DNS server even for offline hosts. If you want to use a specific DNS server, you can add the **--dns-servers DNS_SERVER** option

Answer the questions below

We want Nmap to issue a reverse DNS lookup for all the possibles hosts on a subnet, hoping to get some insights from the names. What option should we add?

> *-R*

Task 9 Summary

You have learned how ARP, ICMP, TCP, and UDP can detect live hosts by completing this room. Any response from a host is an indication that it is online. Below is a quick summary of the command-line options for Nmap that we have covered.

Scan Type	Example Command
ARP Scan	sudo nmap -PR -sn MACHINE_IP/24
ICMP Echo Scan	sudo nmap -PE -sn MACHINE_IP/24
ICMP Timestamp Scan	sudo nmap -PP -sn MACHINE_IP/24
ICMP Address Mask Scan	sudo nmap -PM -sn MACHINE_IP/24
TCP SYN Ping Scan	sudo nmap -PS22,80,443 -sn MACHINE_IP/30
TCP ACK Ping Scan	sudo nmap -PA22,80,443 -sn MACHINE_IP/30
UDP Ping Scan	sudo nmap -PU53,161,162 -sn MACHINE_IP/30

Remember to add **-sn** if you are only interested in host discovery without port-scanning. Omitting -sn will let Nmap default to port-scanning the live hosts.

Option	Purpose
-n	no DNS lookup
-R	reverse-DNS lookup for all hosts
-sn	host discovery only

Answer the questions below

Ensure you have taken note of all the Nmap options explained in this room. To continue learning about Nmap, please join the room Nmap Basic Port Scans, which introduces the basic types of port scans.

> *No answer needed*