Nmap Advance Port Scan

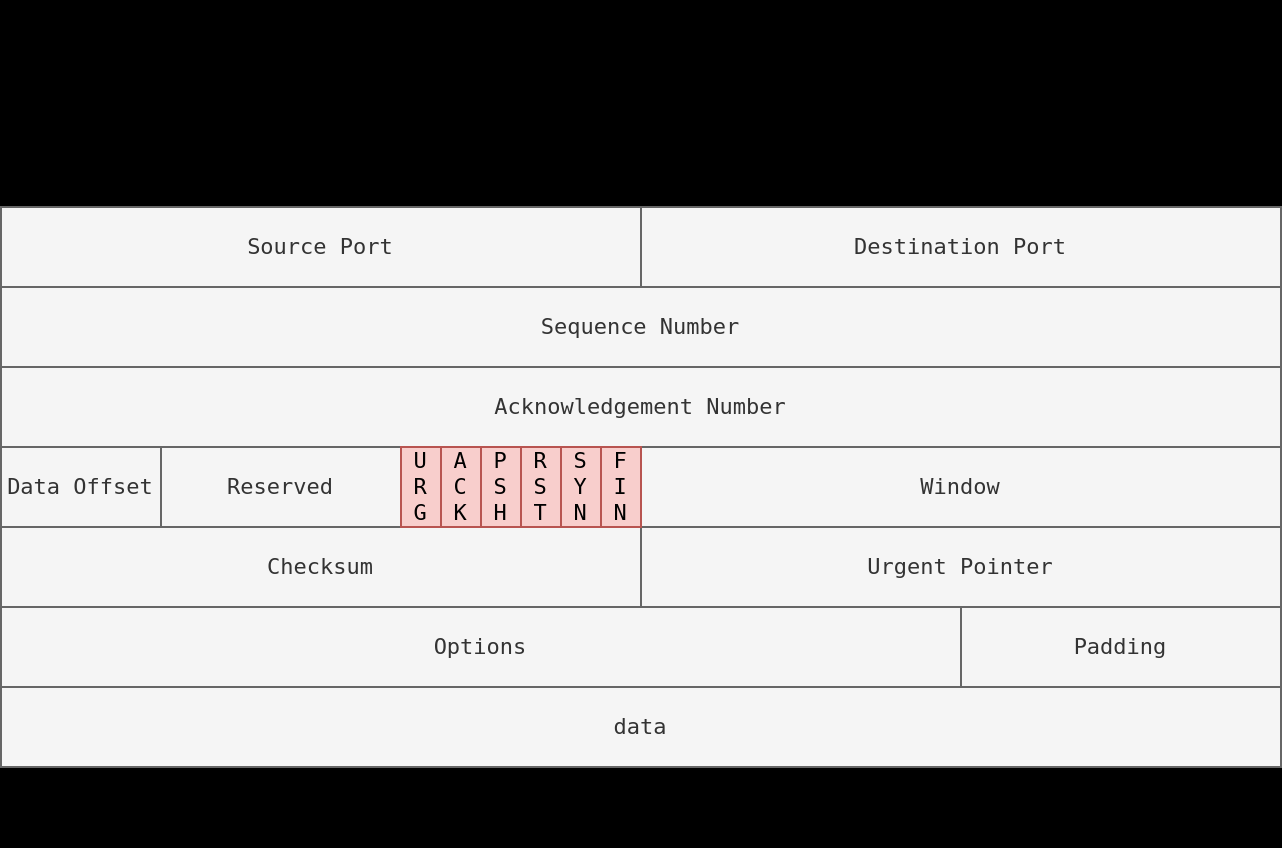
**Task 1 – Introduction**

This room is third in the Nmap series (part of the introduction to Network Security module). In the first two rooms, we learned about live host discovery and basic port scan.

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

In Nmap Basic Port Scans, we covered TCP flags and reviewed the TCP 3-way handshake. To initiate a connection, TCP requires the first packet to have the SYN flag set. Consequently, we can tell if a TCP port is open based on the response we receive.

Security researchers and hackers contemplated the TCP flags, shown in the figure below and explained in the previous room, and started to experiment. They wanted to know what would happen if we send a TCP packet, which is not part of any ongoing TCP connection, with one or more flags set.



For instance, an ACK flag is set when you want to acknowledge received data. An ACK scan is like trying to acknowledge data that was neither sent nor received in the first place. Consider this simple analogy, someone coming to you out of nowhere to tell you, “yes, I hear you, please continue.” when you haven’t said anything.

This room explains advanced types of scans and scan options. Some of these scan types can be useful against specific systems, while others are useful in particular network setups. We will cover the following types of port scans:

* Null scan
* FIN Scan
* Xmas Scan
* Mainmon Scan
* ACK Scan
* Windows scan
* Custom Scan

More, we will cover the following:

* Spoofing IP
* Spoofing MAC
* Decoy Scan
* Fragmented Packet
* Idle/Zombie Scan

**Task 2 - TCP Null Scan, FIN Scan, and Xmas Scan**

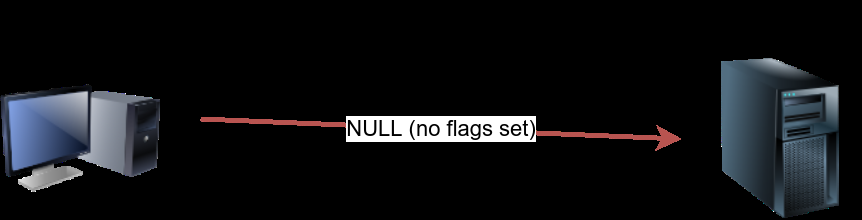
Let’s start with the following three type of scan:

* Null Scan
* Fin Scan
* Xmas Scan

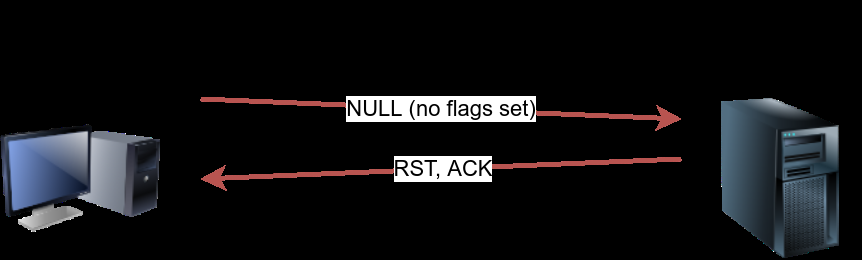
Null Scan

The null scan does not set any flag; all six flag bits are set to zero. You can choose this scan using the -sN option. A TCP packet with no flags set will not trigger any response when it reaches an open port, as shown in the figure below. Therefore, from Nmap’s perspective, a lack of reply in a null scan indicates that either the port is open or a firewall is blocking the packet.

**Nmap -sN TARGET**



However, we expect the target server to respond with an RST packet if the port is closed. Consequently, we can use the lack of RST response to figure out the ports that are not closed: open or filtered.



Below is an example of a null scan against a Linux server. The null scan we carried out has successfully identified the six open ports on the target system. Because the null scan relies on the lack of a response to infer that the port is not closed, it cannot indicate with certainty that these ports are open; there is a possibility that the ports are not responding due to a firewall rule.

pentester@TryHackMe$ sudo nmap -sN MACHINE\_IP

Starting Nmap 7.60 ( https://nmap.org ) at 2021-08-30 10:30 BST

Nmap scan report for 10.10.225.31

Host is up (0.00066s latency).

Not shown: 994 closed ports

PORT STATE SERVICE

22/tcp open|filtered ssh

25/tcp open|filtered smtp

80/tcp open|filtered http

110/tcp open|filtered pop3

111/tcp open|filtered rpcbind

143/tcp open|filtered imap

MAC Address: 02:45:BF:8A:2D:6B (Unknown)

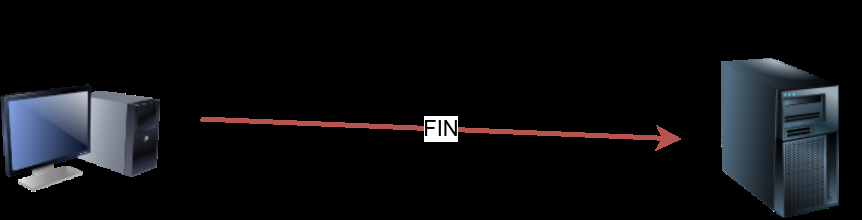
Nmap done: 1 IP address (1 host up) scanned in 96.50 seconds

Note that many Nmap options require root privileges. Unless you are running Nmap as root, you need to use sudo as in the example above using the -sN option.

**FIN Scan**

The FIN scan sends a TCP packet with the FIN flag set. You can choose this scan type using the -sF option. Similarly, no response will be sent if the TCP port is open. Again, Nmap cannot be sure if the port is open or if a firewall is blocking the traffic related to this TCP port.

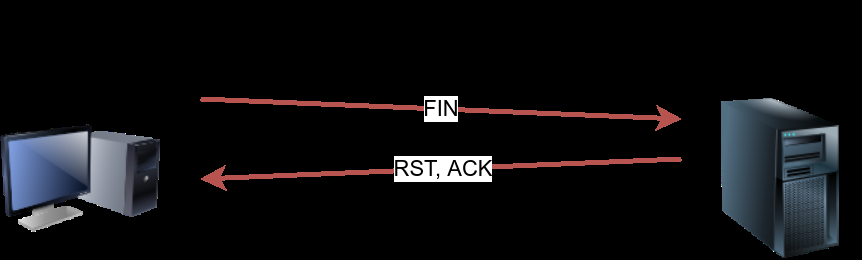
**Nmap -sF TARGET**



**Case: TCP port is open**

However, the target system should respond with an RST if the port is closed. Consequently, we will be able to know which ports are closed and use this knowledge to infer the ports that are open or filtered. It's worth noting some firewalls will 'silently' drop the traffic without sending an RST.

Namp -sF TARGET

 **Case: TCP port is closed.**

Below is an example of a FIN scan against a Linux server. The result is quite similar to the result we obtained earlier using a null scan.

pentester@TryHackMe$ sudo nmap -sF MACHINE\_IP

22/tcp open|filtered ssh

25/tcp open|filtered smtp

80/tcp open|filtered http

110/tcp open|filtered pop3

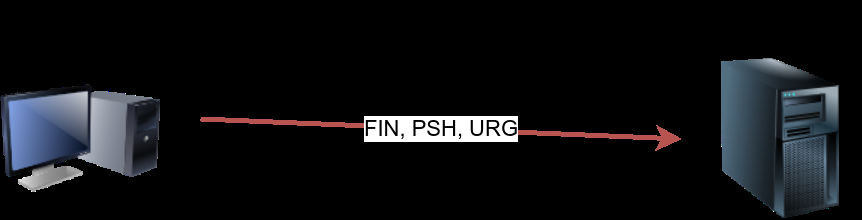
**Xmas Scan**

The Xmas scan gets its name after Christmas tree lights. An Xmas scan sets the FIN, PSH, and URG flags simultaneously. You can select Xmas scan with the option -sX.

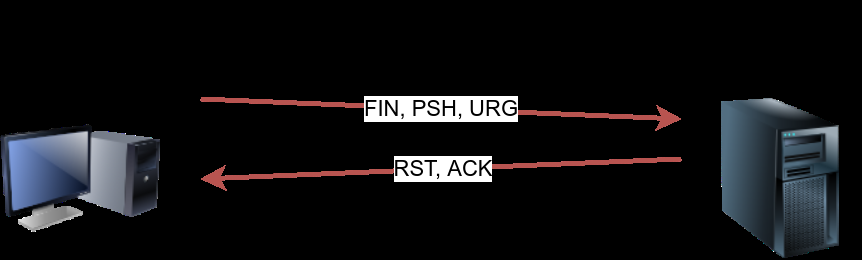
Like the Null scan and FIN scan, if an RST packet is received, it means that the port is closed. Otherwise, it will be reported as open|filtered.

The following two figures show the case when the TCP port is open and the case when the TCP port is closed.

Namp -sX TARGET



**Case: TCP port is open**



**Case: TCP port is closed**

The console output below shows an example of a Xmas scan against a Linux server. The obtained results are pretty similar to that of the null scan and the FIN scan.

pentester@TryHackMe$ sudo nmap -sX MACHINE\_IP

Starting Nmap 7.60 ( https://nmap.org ) at 2021-08-30 10:34 BST

Nmap scan report for 10.10.225.31

Host is up (0.00087s latency).

Not shown: 994 closed ports

PORT STATE SERVICE

22/tcp open|filtered ssh

25/tcp open|filtered smtp

80/tcp open|filtered http

110/tcp open|filtered pop3

111/tcp open|filtered rpcbind

143/tcp open|filtered imap

MAC Address: 02:45:BF:8A:2D:6B (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 84.85 second

**Answer the question:**

1 – In null scan, how many flags are set to 1?

Ans – 0 # because null flag does not set any flag

2 – In a FIN scan, how many flags are set to 1?

Ans - 1 # because FIN scan sends a TCP packet with FIN flag set.

3 – In Xmas scan, how many flags are set to 1?

Ans – 3 #Xmas scan sets the FIN, PSH, and URG flags

4 - Start the VM and load the AttackBox. Once both are ready, open the terminal on the AttackBox and use nmap to launch a FIN scan against the target VM. How many ports appear as open|filtered?

Ans- 7

5 - Repeat your scan launching a null scan against the target VM. How many ports appear as open|filtered?

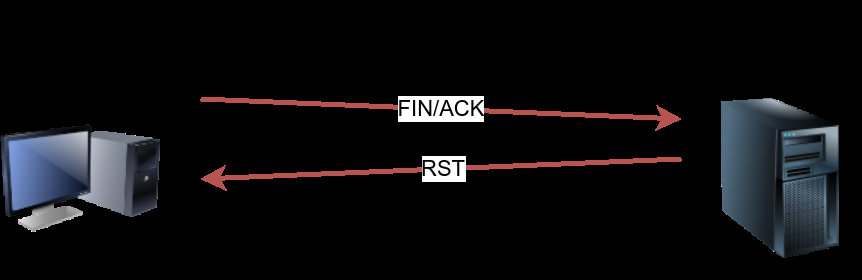
Ans - 7

**Task 3 – TCP Maimon Scan**

Uriel Maimon first described this scan in 1996. In this scan, the FIN and ACK bits are set. The target should send an RST packet as a response. However, certain BSD-derived systems drop the packet if it is an open port exposing the open ports. This scan won’t work on most targets encountered in modern networks; however, we include it in this room to better understand the port scanning mechanism and the hacking mindset. To select this scan type, use the -sM option.

Most target systems respond with an RST packet regardless of whether the TCP port is open. In such a case, we won’t be able to discover the open ports. The figure below shows the expected behaviour in the cases of both open and closed TCP ports.

**Nmap -sM TARGET**



Here is the example command of a TCP Maimon scan against a Linux server.

pentester@TryHackMe$ sudo nmap -sM 10.10.252.27

Starting Nmap 7.60 ( https://nmap.org ) at 2021-08-30 10:36 BST

Nmap scan report for ip-10-10-252-27.eu-west-1.compute.internal (10.10.252.27)

Host is up (0.00095s latency).

All 1000 scanned ports on ip-10-10-252-27.eu-west-1.compute.internal (10.10.252.27) are closed

MAC Address: 02:45:BF:8A:2D:6B (Unknown)

**Answer the question:**

In the Maimon scan, how many flags are set?

Ans - 2

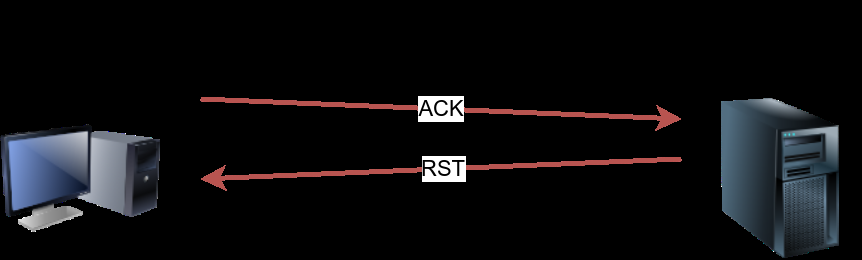
**Task 4 – TCP ACK, Windows and Custom scan**

This task will cover how to perform a TCP ACK scan, a TCP windows scan and how to create your custom flag scan.

TCP ACK Scan

Let’s start with the TCP ACK scan. As the name implies, an ACK scan will send a TCP packet with the ACK flag set. Use the -sA option to choose this scan. As we show in the figure below, the target would respond to the ACK with RST regardless of the state of the port. This behaviour happens because a TCP packet with the ACK flag set should be sent only in response to a received TCP packet to acknowledge the receipt of some data, unlike our case. Hence, this scan won’t tell us whether the target port is open in a simple setup.

**Nmap -sA TARGET**



**Case: TCP port is open or closed**

└─# nmap -sA 10.10.179.113

Starting Nmap 7.91 ( https://nmap.org ) at 2021-10-29 07:15 EDT

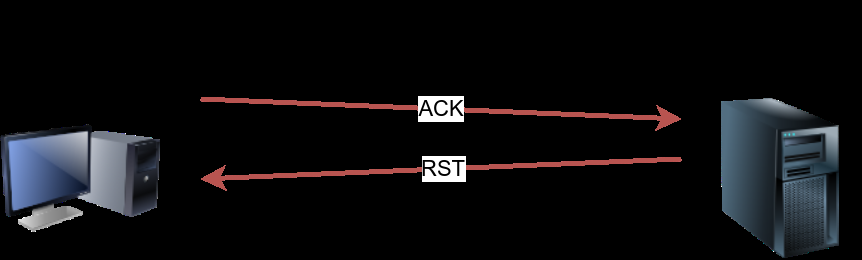
This kind of scan would be helpful if there is a firewall in front of the target. Consequently, based on which ACK packets resulted in responses, you will learn which ports were not blocked by the firewall. In other words, this type of scan is more suitable to discover firewall rule sets and configuration.

After setting up the target VM 10.10.179.113 with a firewall, we repeated the ACK scan. This time, we received some interesting results. As seen in the console output below, we have three ports that aren't being blocked by the firewall. This result indicates that the firewall is blocking all other ports except for these three ports.

**Windows Scan**

Another similar scan is the TCP window scan. The TCP window scan is almost the same as the ACK scan; however, it examines the TCP Window field of the RST packets returned. On specific systems, this can reveal that the port is open. You can select this scan type with the option -sW. As shown in the figure below, we expect to get an RST packet in reply to our “uninvited” ACK packets, regardless of whether the port is open or closed.

**Namp -sW TARGET**



**Case: TCP port is open or closed**

Similarly, launching a TCP window scan against a Linux system with no firewall will not provide much information. As we can see in the console output below, the results of the window scan against a Linux server with no firewall didn’t give any extra information compared to the ACK scan executed earlier.

pentester@TryHackMe$ sudo nmap -sW MACHINE\_IP

Starting Nmap 7.60 ( https://nmap.org ) at 2021-08-30 10:38 BST

Nmap scan report for 10.10.179.113

Host is up (0.0011s latency).

All 1000 scanned ports on ip-10-10-252-27.eu-west-1.compute.internal (10.10.252.27) are closed

MAC Address: 02:45:BF:8A:2D:6B (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.60 seconds

However, as you would expect, if we repeat our TCP window scan against a server behind a firewall, we expect to get more satisfying results. In the console output shown below, the TCP window scan pointed that three ports are detected as closed. (This is in contrast with the ACK scan that labelled the same three ports as unfiltered.) Although we know that these three ports are not closed, we realize they responded differently, indicating that the firewall does not block them.

pentester@TryHackMe$ sudo nmap -sW MACHINE\_IP

Starting Nmap 7.60 ( https://nmap.org ) at 2021-09-07 11:39 BST

Nmap scan report for 10.10.179.113

Host is up (0.00040s latency).

Not shown: 997 filtered ports

PORT STATE SERVICE

22/tcp closed ssh

25/tcp closed smtp

80/tcp closed http

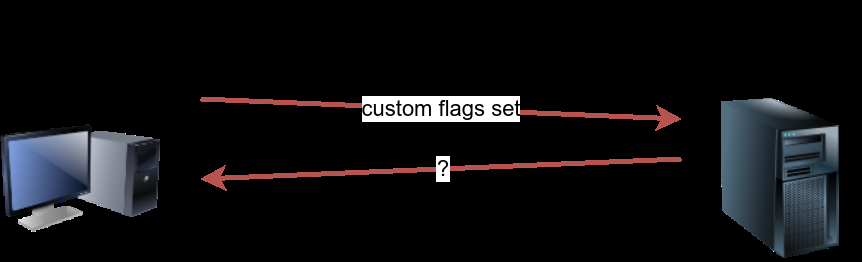
MAC Address: 02:78:C0:D0:4E:E9 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 14.84 seconds

**Custom Scan**

If you want to experiment with a new TCP flag combination beyond the built-in TCP scan types, you can do so using --scanflags. For instance, if you want to set SYN, RST, and FIN simultaneously, you can do so using --scanflags RSTSYNFIN. As shown in the figure below, if you develop your custom scan, you need to know how the different ports will behave to interpret the results in different scenarios correctly.

**Nmap –scanflag CUSTOM\_FLAGS TARGET**



Finally, it is essential to note that the ACK scan and the window scan were very efficient at helping us map out the firewall rules. However, it is vital to remember that just because a firewall is not blocking a specific port, it does not necessarily mean that a service is listening on that port. For example, there is a possibility that the firewall rules need to be updated to reflect recent service changes. Hence, ACK and window scans are exposing the firewall rules, not the services.

**Answer the question:**

`1 – In TCP Windows scan how many flag are set?

Ans – 1 # it examines the TCP Window field of the RST packets returned

2 - You decided to experiment with a custom TCP scan that has the reset flag set. What would you add after --scanflags?

Ans - RST

3 - open the terminal on the AttackBox and use Nmap to launch an ACK scan against the target VM. How many ports appear unfiltered?

Ans – 4 # nmap -sA 10.10.179.113

4 – What is the new port number that happened?

Ans - 443

5 – Is there any service behind the newly discovered port number? (Y/N)?

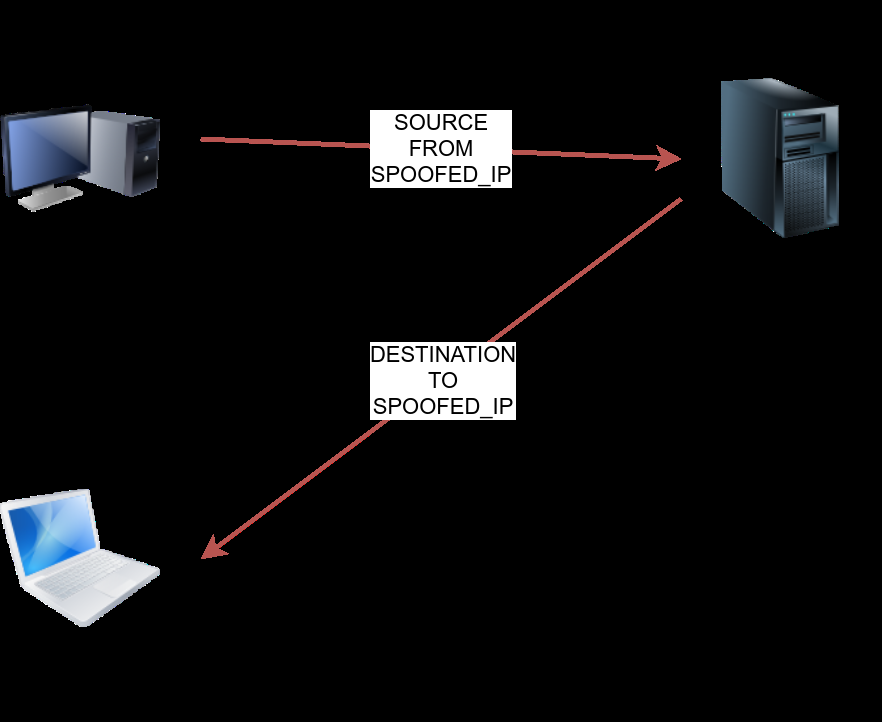
Ans - N

**Task 5 – Spoofing and Decoys**

In some network setups, you will be able to scan a target system using a spoofed IP address and even a spoofed MAC address. Such a scan is only beneficial in a situation where you can guarantee to capture the response. If you try to scan a target from some random network using a spoofed IP address, chances are you won’t have any response routed to you, and the scan results could be unreliable.

The following figure shows the attacker launching the command nmap -S SPOOFED\_IP MACHINE\_IP. Consequently, Nmap will craft all the packets using the provided source IP address SPOOFED\_IP. The target machine will respond to the incoming packets sending the replies to the destination IP address SPOOFED\_IP. For this scan to work and give accurate results, the attacker needs to monitor the network traffic to analyze the replies.

**Nmap -S SPOOFED\_IP MACHINE\_IP**



**Using SPOOFED\_IP as the source IP address for the scan.**

In brief, scanning with a spoofed IP address is there steps:

1. Attacker sends a packet with a spoofed source IP address to the target machine.
2. Target machine replies to the spoofed IP address as the destination.
3. Attacker captures the replies to figure out open ports.

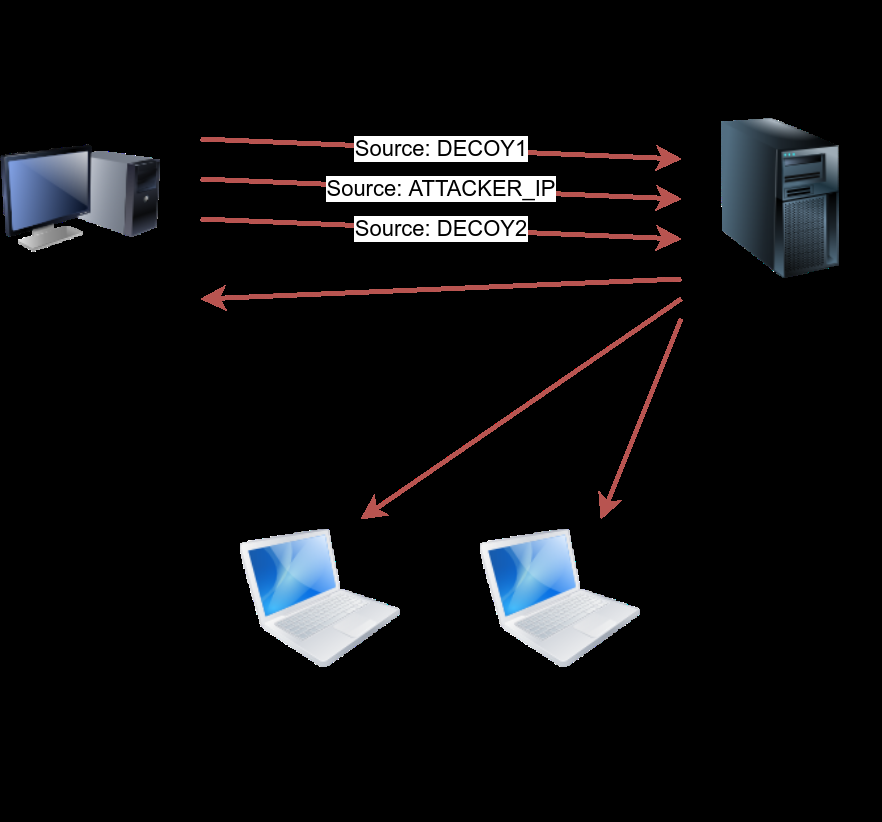
In general, you expect to specify the network interface using -e and to explicitly disable ping scan -Pn. Therefore, instead of nmap -S SPOOFED\_IP MACHINE\_IP, you will need to issue nmap -e NET\_INTERFACE -Pn -S SPOOFED\_IP MACHINE\_IP to tell Nmap explicitly which network interface to use and not to expect to receive a ping reply. It is worth repeating that this scan will be useless if the attacker system cannot monitor the network for responses.

When you are on the same subnet as the target machine, you would be able to spoof your MAC address as well. You can specify the source MAC address using

--spoof-mac SPOOFED\_MAC. This address spoofing is only possible if the attacker and the target machine are on the same Ethernet (802.3) network or same WiFi (802.11).

Spoofing only works in a minimal number of cases where certain conditions are met. Therefore, the attacker might resort to using decoys to make it more challenging to be pinpointed. The concept is simple, make the scan appears to be coming from many IP addresses so that the attacker’s IP address would be lost among them. As we see in the figure below, the scan of the target machine will appear to be coming from 3 different sources, and consequently, the replies will go the decoys as well.

**Nmap -D DECOY1,ME,DECOY2 MACINE\_IP**



You can launch a decoy scan by specifying a specific or random IP address after -D. For example, nmap -D 10.10.0.1,10.10.0.2,ME MACHINE\_IP will make the scan of MACHINE\_IP appear as coming from the IP addresses 10.10.0.1, 10.10.0.2, and then ME to indicate that your IP address should appear in the third order. Another example command would be

nmap -D 10.10.0.1,10.10.0.2,RND,RND,ME MACHINE\_IP, where the third and fourth source IP addresses are assigned randomly, while the fifth source is going to be the attacker’s IP address. In other words, each time you execute the latter command, you would expect two new random IP addresses to be the third and fourth decoy sources.

**Answer the question:**

`1 - What do you need to add to the command sudo nmap MACHINE\_IP to make the scan appear as if coming from the source IP address 10.10.10.11 instead of your IP address?

Ans -: -S 10.10.10.11

2 - What do you need to add to the command sudo nmap 10.10.211.226 to make the scan appear as if coming from the source IP addresses 10.10.20.21 and 10.10.20.28 in addition to your IP address?

Ans -: -D 10.10.20.21,10.10.20.28,ME

**Fragmented Packets**

**Firewall**

A firewall is a piece of software or hardware that permits packets to pass through or blocks them. It functions based on firewall rules, summarized as blocking all traffic with exceptions or allowing all traffic with exceptions. For instance, you might block all traffic to your server except those coming to your web server. A traditional firewall inspects, at least, the IP header and the transport layer header. A more sophisticated firewall would also try to examine the data carried by the transport layer.

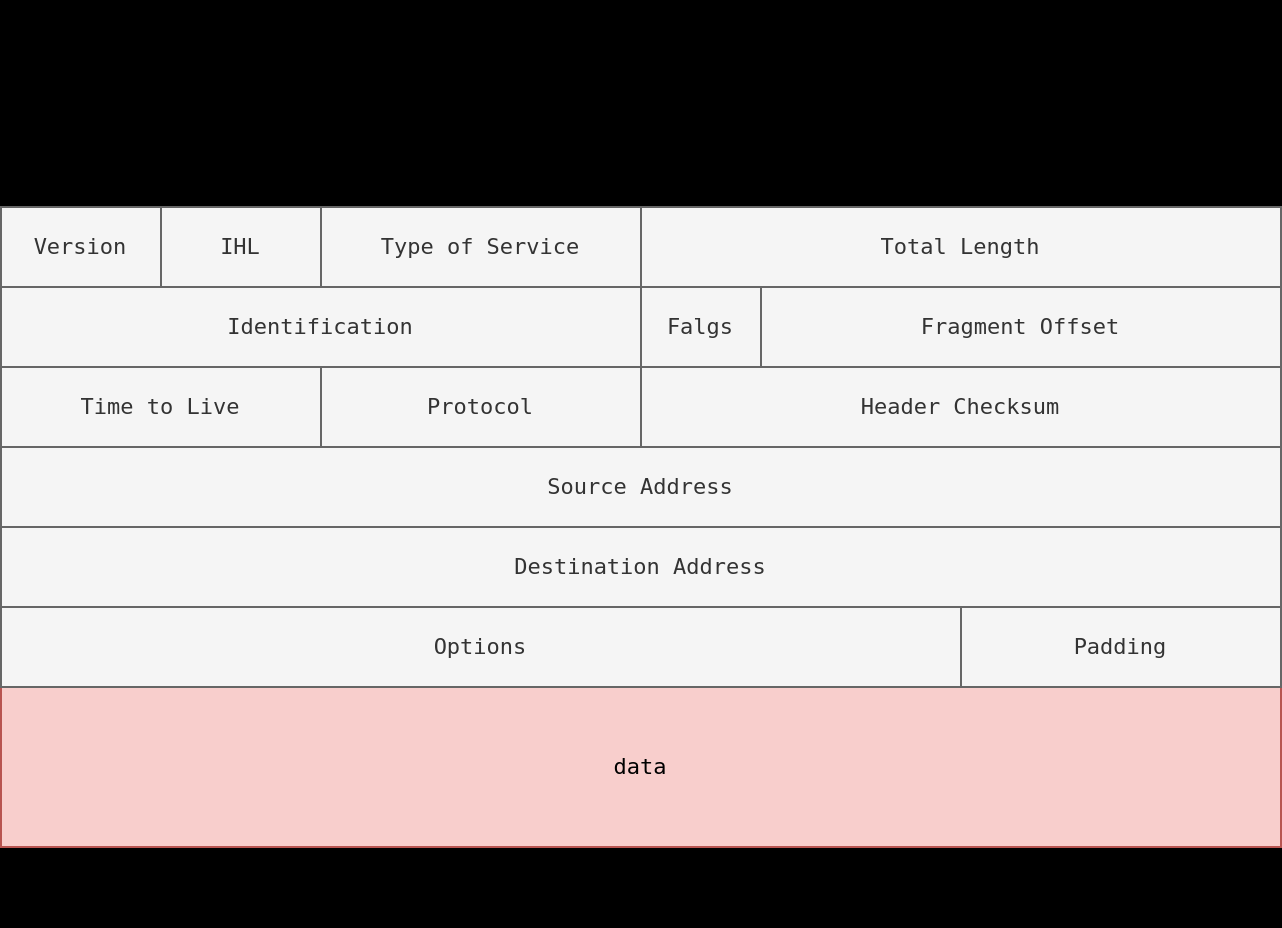
**IDS**

An intrusion detection system (IDS) inspects network packets for select behavioural patterns or specific content signatures. It raises an alert whenever a malicious rule is met. In addition to the IP header and transport layer header, an IDS would inspect the data contents in the transport layer and check if it matches any malicious patterns. How can you make it less likely for a traditional firewall/IDS to detect your Nmap activity? It is not easy to answer this; however, depending on the type of firewall/IDS, you might benefit from dividing the packet into smaller packets.

**Fragmented Packets**

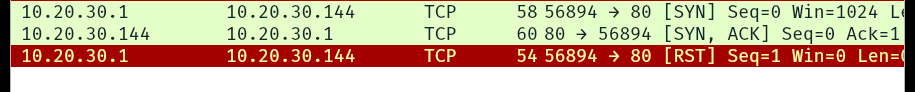
Nmap provides the option -f to fragment packets. Once chosen, the IP data will be divided into 8 bytes or less. Adding another -f (-f -f or -ff) will split the data into 16 byte-fragments instead of 8. You can change the default value by using the --mtu; however, you should always choose a multiple of 8.

To properly understand fragmentation, we need to look at the IP header in the figure below. It might look complicated at first, but we notice that we know most of its fields. In particular, notice the source address taking 32 bits (4 bytes) on the fourth row, while the destination address is taking another 4 bytes on the fifth row. The data that we will fragment across multiple packets is highlighted in red. To aid in the reassembly on the recipient side, IP uses the identification (ID) and fragment offset, shown on the second row of the figure below.

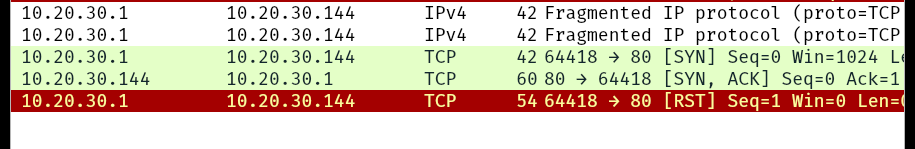


Let’s compare running sudo nmap -sS -p80 10.20.30.144 and sudo nmap -sS -p80 -f 10.20.30.144. As you know by now, this will use stealth TCP SYN scan on port 80; however, in the second command, we are requesting Nmap to fragment the IP packets.

In the first two lines, we can see an ARP query and response. Nmap issued an ARP query because the target is on the same Ethernet. The second two lines show a TCP SYN ping and a reply. The fifth line is the beginning of the port scan; Nmap sends a TCP SYN packet to port 80. In this case, the IP header is 20 bytes, and the TCP header is 24 bytes. Note that the minimum size of the TCP header is 20 bytes.



With fragmentation requested via -f, the 24 bytes of the TCP header will be divided into multiples of 8 bytes, with the last fragment containing 8 bytes or less of the TCP header. Since 24 is divisible by 8, we got 3 IP fragments; each has 20 bytes of IP header and 8 bytes of TCP header. We can see the three fragments between the fifth and the seventh lines.



Note that if you added -ff (or -f -f), the fragmentation of the data will be multiples of 16. In other words, the 24 bytes of the TCP header, in this case, would be divided over two IP fragments, the first containing 16 bytes and the second containing 8 bytes of the TCP header.

On the other hand, if you prefer to increase the size of your packets to make them look innocuous, you can use the option --data-length NUM, where num specifies the number of bytes you want to append to your packets.

**Answer the question:**

If the TCP segments has a size of 64, and -ff options is being used, how many IP fragments will you get?

Ans - 4

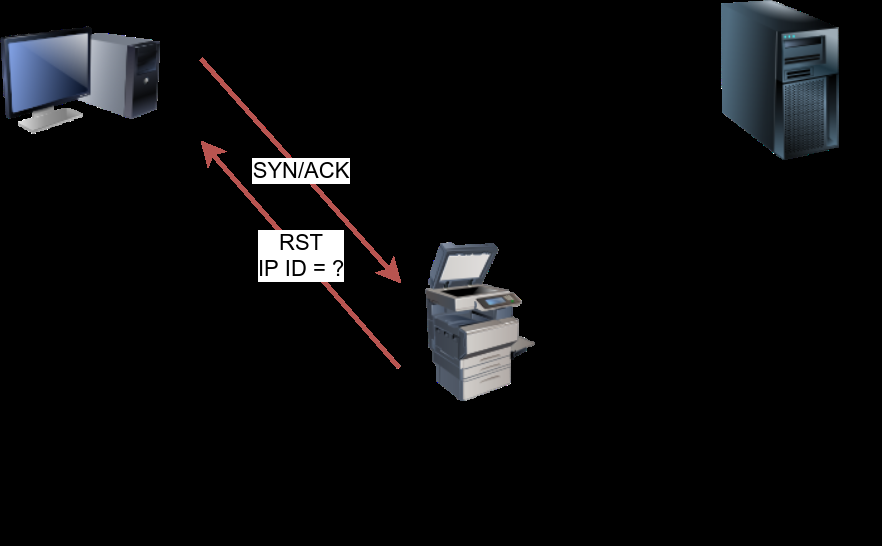
**Task 7 – Idle / Zombie Scan**

Spoofing the source IP address can be a great approach to scanning stealthily. However, spoofing will only work in specific network setups. It requires you to be in a position where you can monitor the traffic. Considering these limitations, spoofing your IP address can have little use; however, we can give it an upgrade with the idle scan.

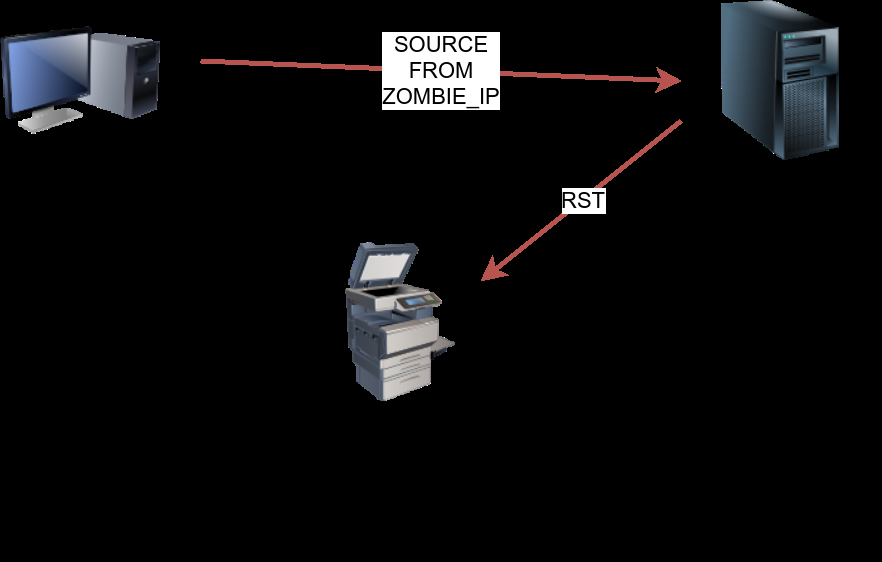
The idle scan, or zombie scan, requires an idle system connected to the network that you can communicate with. Practically, Nmap will make each probe appear as if coming from the idle (zombie) host, then it will check for indicators whether the idle (zombie) host received any response to the spoofed probe. This is accomplished by checking the IP identification (IP ID) value in the IP header. You can run an idle scan using nmap -sI ZOMBIE\_IP 10.10.211.226, where ZOMBIE\_IP is the IP address of the idle host (zombie).

The idle (zombie) scan require the following three steps to discover whether a port is open.

1. Trigger the idle host to respond so that you can record the current IP ID on the idle host.
2. Send a SYN packet to a TCP port on the target. The packet should be spoofed to appear as if it was coming from the idle host (zombie) IP address.
3. Trigger the idle machine again to respond so that you can compare the new IP ID with the one received earlier.



The attacker will send a SYN packet to the TCP port they want to check on the target machine in the next step. However, this packet will use the idle host (zombie) IP address as the source. Three scenarios would arise. In the first scenario, shown in the figure below, the TCP port is closed; therefore, the target machine responds to the idle host with an RST packet. The idle host does not respond; hence its IP ID is not incremented.



**Attacker system sends to target machine a SYN packet spoofed as sent by the idle system. Case: Port is Open**

In the third scenario, the target machine does not respond at all due to firewall rules. This lack of response will lead to the same result as with the closed port; the idle host won’t increase the IP ID.

For the final step, the attacker sends another SYN/ACK to the idle host. The idle host responds with an RST packet, incrementing the IP ID by one again. The attacker needs to compare the IP ID of the RST packet received in the first step with the IP ID of the RST packet received in this third step. If the difference is 1, it means the port on the target machine was closed or filtered. However, if the difference is 2, it means that the port on the target was open.

It is worth repeating that this scan is called an idle scan because choosing an idle host is indispensable for the accuracy of the scan. If the “idle host” is busy, all the returned IP IDs would be useless.

**Answer the question:**

You discovered a rarely-used network printer with the IP address 10.10.5.5, and you decide to use it as a zombie in your idle scan. What argument should you add to your Nmap command? Ans – *nmap -sI 10.10.5.5*

**Task 8 – Getting More Details**

You might consider adding --reason if you want Nmap to provide more details regarding its reasoning and conclusions. Consider the two scans below to the system; however, the latter adds --reason.

pentester@TryHackMe$ sudo nmap -sS 10.10.252.27

Starting Nmap 7.60 ( https://nmap.org ) at 2021-08-30 10:39 BST

Nmap scan report for ip-10-10-252-27.eu-west-1.compute.internal (10.10.252.27)

Host is up (0.0020s latency).

Not shown: 994 closed ports

PORT STATE SERVICE

22/tcp open ssh

25/tcp open smtp

80/tcp open http

110/tcp open pop3

111/tcp open rpcbind

143/tcp open imap

MAC Address: 02:45:BF:8A:2D:6B (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.60 seconds

----------------------------------------------------------------------------------------------------------------

pentester@TryHackMe$ sudo nmap -sS --reason 10.10.252.27

Host is up, received arp-response (0.0020s latency).

Not shown: 994 closed ports

Reason: 994 resets

PORT STATE SERVICE REASON

22/tcp open ssh syn-ack ttl 64

25/tcp open smtp syn-ack ttl 64

80/tcp open http syn-ack ttl 64

110/tcp open pop3 syn-ack ttl 64

111/tcp open rpcbind syn-ack ttl 64

143/tcp open imap syn-ack ttl 64

Providing the --reason flag gives us the explicit reason why Nmap concluded that the system is up or a particular port is open. In this console output above, we can see that this system is considered online because Nmap “received arp-response.” On the other hand, we know that the SSH port is deemed to be open because Nmap received a “syn-ack” packet back.

For more detailed output, you can consider using -v for verbose output or -vv for even more verbosity.

pentester@TryHackMe$ sudo nmap -sS -vv 10.10.252.27

Starting Nmap 7.60 ( https://nmap.org ) at 2021-08-30 10:41 BST

Initiating ARP Ping Scan at 10:41

Scanning 10.10.252.27 [1 port]

Completed ARP Ping Scan at 10:41, 0.22s elapsed (1 total hosts)

Initiating Parallel DNS resolution of 1 host. at 10:41

Completed Parallel DNS resolution of 1 host. at 10:41, 0.00s elapsed

Initiating SYN Stealth Scan at 10:41

Scanning ip-10-10-252-27.eu-west-1.compute.internal (10.10.252.27) [1000 ports]

Discovered open port 22/tcp on 10.10.252.27

Discovered open port 25/tcp on 10.10.252.27

Discovered open port 80/tcp on 10.10.252.27

Discovered open port 110/tcp on 10.10.252.27

Discovered open port 111/tcp on 10.10.252.27

Discovered open port 143/tcp on 10.10.252.27

Completed SYN Stealth Scan at 10:41, 1.25s elapsed (1000 total ports)

Nmap scan report for ip-10-10-252-27.eu-west-1.compute.internal (10.10.252.27)

Host is up, received arp-response (0.0019s latency).

Scanned at 2021-08-30 10:41:02 BST for 1s

Not shown: 994 closed ports

Reason: 994 resets

PORT STATE SERVICE REASON

22/tcp open ssh syn-ack ttl 64

25/tcp open smtp syn-ack ttl 64

80/tcp open http syn-ack ttl 64

110/tcp open pop3 syn-ack ttl 64

111/tcp open rpcbind syn-ack ttl 64

143/tcp open imap syn-ack ttl 64

MAC Address: 02:45:BF:8A:2D:6B (Unknown)

Read data files from: /usr/bin/../share/nmap

Nmap done: 1 IP address (1 host up) scanned in 1.59 seconds

Raw packets sent: 1002 (44.072KB) | Rcvd: 1002 (40.092KB)

If -vv does not satisfy your curiosity, you can use -d for debugging details or -dd for even more details. You can guarantee that using -d will create an output that extends beyond a single screen.

**Answer the question:**

Launch the AttackBox if you haven't done so already. After you make sure that you have terminated the VM from Task 4, start the VM for this task. Wait for it to load completely, then open the terminal on the AttackBox and use Nmap with nmap -sS -F --reason 10.10.211.226 to scan the VM. What is the reason provided for the stated port(s) being open?

Ans- syn-ack

**Task 9 – Summary**

This room covered following types of scans.

|  |  |
| --- | --- |
| **Port Scan Type** | **Example Command** |
| TCP Null Scan | sudo nmap -sN 10.10.211.226 |
| TCP FIN Scan | sudo nmap -sF 10.10.211.226 |
| TCP Xmas Scan | sudo nmap -sX 10.10.211.226 |
| TCP Maimon Scan | sudo nmap -sM 10.10.211.226 |
| TCP ACK Scan | sudo nmap -sA 10.10.211.226 |
| TCP Windows Scan | sudo nmap -sW 10.10.211.226 |
| Custom TCP Scan | sudo nmap --scanflags URGACKPSHRSTSYNFIN 10.10.211.226 |
| Spoofed Source IP | sudo nmap -S SPOOFED\_IP 10.10.211.226 |
| Spoofed source Address | --spoof-mac SPOOFED\_MAC |
| Decoy Scan | nmap -D DECOY\_IP,ME 10.10.211.226 |
| Idle(zombie) Scan | sudo nmap -sI ZOMBIE\_IP 10.10.211.226 |
| Fragment IP data into 8 bytes | -f |
| Fragment IP data into 16 bytes | -ff |

These scan type rely on settings TCP flags in unexpected ways to prompt ports for a reply. Null, FIN, and Xmas scan provoke a response from closed ports, while Maimon, Ack, and Windows scans provoke a response from open and closed ports.

|  |  |
| --- | --- |
| Options | Purpose |
| --reason | Explain how Nmap made its conclusion |
| -v | Verbose |
| -vv | Very verbose |
| -d | Debugging |
| -dd | More details for debugging |