

Nmap for Pentester: Host Discovery

December 16, 2020 By Raj Chandel

Nmap has become one of the most popular tools in network scanning by leaving other scanners behind. Many times the hosts in some organisations are secured using firewalls or intrusion prevention systems which result in the failure of scanning due to the present set of rules which are used to block network traffic. In Nmap, a pentester can easily make use of alternate host discovery techniques to prevent this from happening. It consists of certain features that make the network traffic a little less suspicious. Hence, let us look at various techniques of Host Discovery.

Table of Content

- **Ping Sweep (No port scan)**
- **Disable-arp-ping**
- **Send-ip**
- **TCP Flags**
- **Types of Scans**
- **TCP SYN Ping Scan**
- **TCP ACK Ping Scan**
- **ICMP ECHO Ping Scan**
- **ICMP ECHO Ping Sweep**
- **UDP Ping Scan**
- **IP Protocol Ping scan**
- **ARP Ping Scan**
- **Traceroute**

Ping Sweep

Let's begin with scanning the entire network by using the Ping sweep scan (-sP).

```
nmap -sP 192.168.1.0/24
```

```
root@kali:~# nmap -sP 192.168.1.0/24
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 05:43 EST
Nmap scan report for dsldevice.lan (192.168.1.1)
Host is up (0.0012s latency).
MAC Address: 18:45:93:69:A5:10 (Taicang T&W Electronics)
Nmap scan report for 192.168.1.3
Host is up (0.00030s latency).
MAC Address: 8C:EC:4B:71:C5:DE (Dell)
Nmap scan report for 192.168.1.4
Host is up (0.024s latency).
MAC Address: 2A:84:98:9F:E5:5E (Unknown)
Nmap scan report for 192.168.1.5
Host is up (0.012s latency).
MAC Address: 30:24:32:1F:89:AC (Intel Corporate)
Nmap scan report for 192.168.1.8
Host is up (0.0058s latency).
MAC Address: 44:CB:8B:C2:20:DA (LG Innotek)
Nmap scan report for 192.168.1.12
Host is up (0.00027s latency).
MAC Address: 00:0C:29:78:20:90 (VMware)
Nmap scan report for 192.168.1.108
Host is up (0.00017s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap scan report for 192.168.1.9
Host is up.
Nmap done: 256 IP addresses (8 hosts up) scanned in 23.67 seconds
root@kali:~#
```

When you closely observe the packets in the Wireshark, you see that here only ARP packets are being sent while scanning the network,

| arp | | | | | | |
|-----|-------------|-------------------|-------------------|----------|--------|--|
| No. | Time | Source | Destination | Protocol | Length | Info |
| 64 | 0.550463087 | TaicangT_69:a5... | Dell_71:c5:de | ARP | 60 | Who has 192.168.1.3? Tell 192.168.1.1 |
| 65 | 0.550463118 | Dell_71:c5:de | TaicangT_69:a5:10 | ARP | 60 | 192.168.1.3 is at 8c:ec:4b:71:c5:de |
| 209 | 1.589157998 | TaicangT_69:a5... | VMware_b2:bb:77 | ARP | 60 | Who has 192.168.1.9? Tell 192.168.1.1 |
| 210 | 1.589181561 | VMware_b2:bb:77 | TaicangT_69:a5:10 | ARP | 42 | 192.168.1.9 is at 00:0c:29:b2:bb:77 |
| 228 | 1.974212283 | VMware_b2:bb:77 | Broadcast | ARP | 42 | Who has 192.168.1.1? Tell 192.168.1.9 |
| 229 | 1.974288490 | VMware_b2:bb:77 | Broadcast | ARP | 42 | Who has 192.168.1.2? Tell 192.168.1.9 |
| 230 | 1.974336247 | VMware_b2:bb:77 | Broadcast | ARP | 42 | Who has 192.168.1.3? Tell 192.168.1.9 |
| 231 | 1.974396043 | VMware_b2:bb:77 | Broadcast | ARP | 42 | Who has 192.168.1.4? Tell 192.168.1.9 |
| 232 | 1.974433312 | VMware_b2:bb:77 | Broadcast | ARP | 42 | Who has 192.168.1.5? Tell 192.168.1.9 |
| 233 | 1.974456184 | Dell_71:c5:de | VMware_b2:bb:77 | ARP | 60 | 192.168.1.3 is at 8c:ec:4b:71:c5:de |
| 234 | 1.974463392 | VMware_b2:bb:77 | Broadcast | ARP | 42 | Who has 192.168.1.6? Tell 192.168.1.9 |
| 235 | 1.974494541 | VMware_b2:bb:77 | Broadcast | ARP | 42 | Who has 192.168.1.7? Tell 192.168.1.9 |
| 236 | 1.974541930 | VMware_b2:bb:77 | Broadcast | ARP | 42 | Who has 192.168.1.8? Tell 192.168.1.9 |
| 237 | 1.974575204 | VMware_b2:bb:77 | Broadcast | ARP | 42 | Who has 192.168.1.10? Tell 192.168.1.9 |
| 238 | 1.974604998 | VMware_b2:bb:77 | Broadcast | ARP | 42 | Who has 192.168.1.11? Tell 192.168.1.9 |

▶ Frame 64: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface eth0, id 0
 ▶ Ethernet II, Src: TaicangT_69:a5:10 (18:45:93:69:a5:10), Dst: Dell_71:c5:de (8c:ec:4b:71:c5:de)
 ▶ Address Resolution Protocol (request)

```

0000  8c ec 4b 71 c5 de 18 45 93 69 a5 10 08 06 00 01  ..Kq...E.i.....
0010  08 00 06 04 00 01 18 45 93 69 a5 10 c0 a8 01 01  .....E.i.....
0020  00 00 00 00 00 00 c0 a8 01 03 00 00 00 00 00 00  .....
0030  00 00 00 00 00 00 00 00 00 00 00 00  .....

```

Note: Working of **-sP** and **-sn** is the same.

Let us try the same by using **the no port scanning (-sn)** option. In this option, we are also **using -packet-trace** option which will enable you to see the detailed packet transfer without making use of Wireshark. Here you can observe the ARP packets being received.

```
nmap -sn 192.168.1.0/24 --packet-trace
```

```

root@kali:~# nmap -sn 192.168.1.0/24 --packet-trace
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 05:48 EST
SENT (0.0687s) ARP who-has 192.168.1.1 tell 192.168.1.9
SENT (0.0688s) ARP who-has 192.168.1.2 tell 192.168.1.9
SENT (0.0689s) ARP who-has 192.168.1.3 tell 192.168.1.9
SENT (0.0690s) ARP who-has 192.168.1.4 tell 192.168.1.9
SENT (0.0691s) ARP who-has 192.168.1.5 tell 192.168.1.9
SENT (0.0692s) ARP who-has 192.168.1.6 tell 192.168.1.9
SENT (0.0692s) ARP who-has 192.168.1.7 tell 192.168.1.9
SENT (0.0693s) ARP who-has 192.168.1.8 tell 192.168.1.9
SENT (0.0694s) ARP who-has 192.168.1.10 tell 192.168.1.9
SENT (0.0695s) ARP who-has 192.168.1.11 tell 192.168.1.9
RCVD (0.0690s) ARP reply 192.168.1.3 is-at 8C:EC:4B:71:C5:DE
RCVD (0.0699s) ARP reply 192.168.1.1 is-at 18:45:93:69:A5:10
SENT (0.0730s) ARP who-has 192.168.1.14 tell 192.168.1.9
SENT (0.0731s) ARP who-has 192.168.1.15 tell 192.168.1.9
SENT (0.0731s) ARP who-has 192.168.1.16 tell 192.168.1.9
SENT (0.0732s) ARP who-has 192.168.1.17 tell 192.168.1.9
RCVD (0.0791s) ARP reply 192.168.1.4 is-at 2A:84:98:9F:E5:5E
RCVD (0.0796s) ARP reply 192.168.1.5 is-at 30:24:32:1F:89:AC
SENT (0.0820s) ARP who-has 192.168.1.20 tell 192.168.1.9
SENT (0.0822s) ARP who-has 192.168.1.21 tell 192.168.1.9
SENT (0.0823s) ARP who-has 192.168.1.22 tell 192.168.1.9
SENT (0.0824s) ARP who-has 192.168.1.23 tell 192.168.1.9
SENT (0.1699s) ARP who-has 192.168.1.26 tell 192.168.1.9
SENT (0.1703s) ARP who-has 192.168.1.27 tell 192.168.1.9
SENT (0.1705s) ARP who-has 192.168.1.28 tell 192.168.1.9
SENT (0.1708s) ARP who-has 192.168.1.29 tell 192.168.1.9
SENT (0.1710s) ARP who-has 192.168.1.30 tell 192.168.1.9
SENT (0.1712s) ARP who-has 192.168.1.31 tell 192.168.1.9

```

Now when we have seen that ARP packets are seen in the network, we will make use of **--disable-arp-ping** option where you can see that there are 4 packets being sent.

Disable-arp-ping

To disable the ARP discovery, Nmap provides this option.

```
nmap -sn 192.168.1.108 --disable-arp-ping
```

```

root@kali:~# nmap -sn 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 05:58 EST
Nmap scan report for 192.168.1.108
Host is up (0.00027s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.12 seconds
root@kali:~#

```

And you will see that the ARP packets are not visible

Note: Scanning Local Network with Nmap where nmap sends an ARP packet with every scan. If an external network is to be scanned; Nmap sends following request packets when **--disable-arp-ping** is used:

ICMP echo request (Type 8)

ICMP timestamp request(Type 13)

TCP SYN to port 443

TCP ACK to port 80

The image shows a Wireshark packet capture on interface eth0. The filter is set to `ip.addr == 192.168.1.108`. The packet list shows several packets:

| No | Tin | Source | Destination | Protocol | Length | Info |
|-----|-----|---------------|---------------|----------|--------|---|
| ... | ... | 192.168.1.9 | 192.168.1.108 | ICMP | 42 | Echo (ping) request id=0x3b18, seq=0/0, ttl=57 (reply expected) |
| ... | ... | 192.168.1.9 | 192.168.1.108 | TCP | 58 | 43181 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460 |
| ... | ... | 192.168.1.9 | 192.168.1.108 | TCP | 54 | 43181 → 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0 |
| ... | ... | 192.168.1.9 | 192.168.1.108 | ICMP | 54 | Timestamp request id=0x4674, seq=0/0, ttl=44 |
| ... | ... | 192.168.1.108 | 192.168.1.9 | ICMP | 60 | Echo (ping) reply id=0x3b18, seq=0/0, ttl=64 (reply received) |
| ... | ... | 192.168.1.108 | 192.168.1.9 | TCP | 60 | 443 → 43181 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0 |
| ... | ... | 192.168.1.108 | 192.168.1.9 | TCP | 60 | 80 → 43181 [RST] Seq=1 Win=0 Len=0 |
| ... | ... | 192.168.1.108 | 192.168.1.9 | ICMP | 60 | Timestamp reply id=0x4674, seq=0/0, ttl=64 |

The packet details pane shows the selected packet (Frame 794) with the following information:

- Frame 794: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface eth0, id 0
- Ethernet II, Src: VMware_b2:bb:77 (00:0c:29:b2:bb:77), Dst: VMware_c8:9c:50 (00:0c:29:c8:9c:50)
- Internet Protocol Version 4, Src: 192.168.1.9, Dst: 192.168.1.108
- Internet Control Message Protocol

The packet bytes pane shows the raw data of the selected packet:

```
0000  00 0c 29 c8 9c 50 00 0c 29 b2 bb 77 08 00 45 00  ..).P..).w..E.
0010  00 1c 6c b8 00 00 39 01 91 63 c0 a8 01 09 c0 a8  ..1..9..c.....
0020  01 6c 08 00 bc e7 3b 18 00 00  ..1....;. ..
```

You can also make use of `--send-ip` option to get the same results as in the step above.

send-ip

```
nmap -sn 192.168.1.108 --packet-trace --send-ip
```



```

root@kali:~# nmap -sn 192.168.1.108 --packet-trace --send-ip
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 05:55 EST
SENT (0.0588s) ICMP [192.168.1.9 > 192.168.1.108 Echo request (type=8/code=0) id=2974
SENT (0.0589s) TCP 192.168.1.9:43573 > 192.168.1.108:443 S ttl=58 id=30850 iplen=44
SENT (0.0589s) TCP 192.168.1.9:43573 > 192.168.1.108:80 A ttl=55 id=52947 iplen=40
SENT (0.0590s) ICMP [192.168.1.9 > 192.168.1.108 Timestamp request (type=13/code=0) id=2974
RCVD (0.0590s) ICMP [192.168.1.108 > 192.168.1.9 Echo reply (type=0/code=0) id=2974
NSOCK INFO [0.1030s] nsock_iod_new2(): nsock_iod_new (IOD #1)
NSOCK INFO [0.1030s] nsock_connect_udp(): UDP connection requested to 192.168.1.1:53
NSOCK INFO [0.1030s] nsock_read(): Read request from IOD #1 [192.168.1.1:53] (timeout)
NSOCK INFO [0.1030s] nsock_write(): Write request for 44 bytes to IOD #1 EID 27 [192.168.1.1:53]
NSOCK INFO [0.1030s] nsock_trace_handler_callback(): Callback: CONNECT SUCCESS for EID 27
NSOCK INFO [0.1030s] nsock_trace_handler_callback(): Callback: WRITE SUCCESS for EID 27
NSOCK INFO [0.1090s] nsock_trace_handler_callback(): Callback: READ SUCCESS for EID 27
NSOCK INFO [0.1090s] nsock_read(): Read request from IOD #1 [192.168.1.1:53] (timeout)
NSOCK INFO [0.1090s] nsock_iod_delete(): nsock_iod_delete (IOD #1)
NSOCK INFO [0.1090s] nevent_delete(): nevent_delete on event #34 (type READ)
Nmap scan report for 192.168.1.108
Host is up (0.00024s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.12 seconds

```

Host Discovery is considered to be the most primary step in Information Gathering which provides accurate results on active ports and IP addresses in a network.

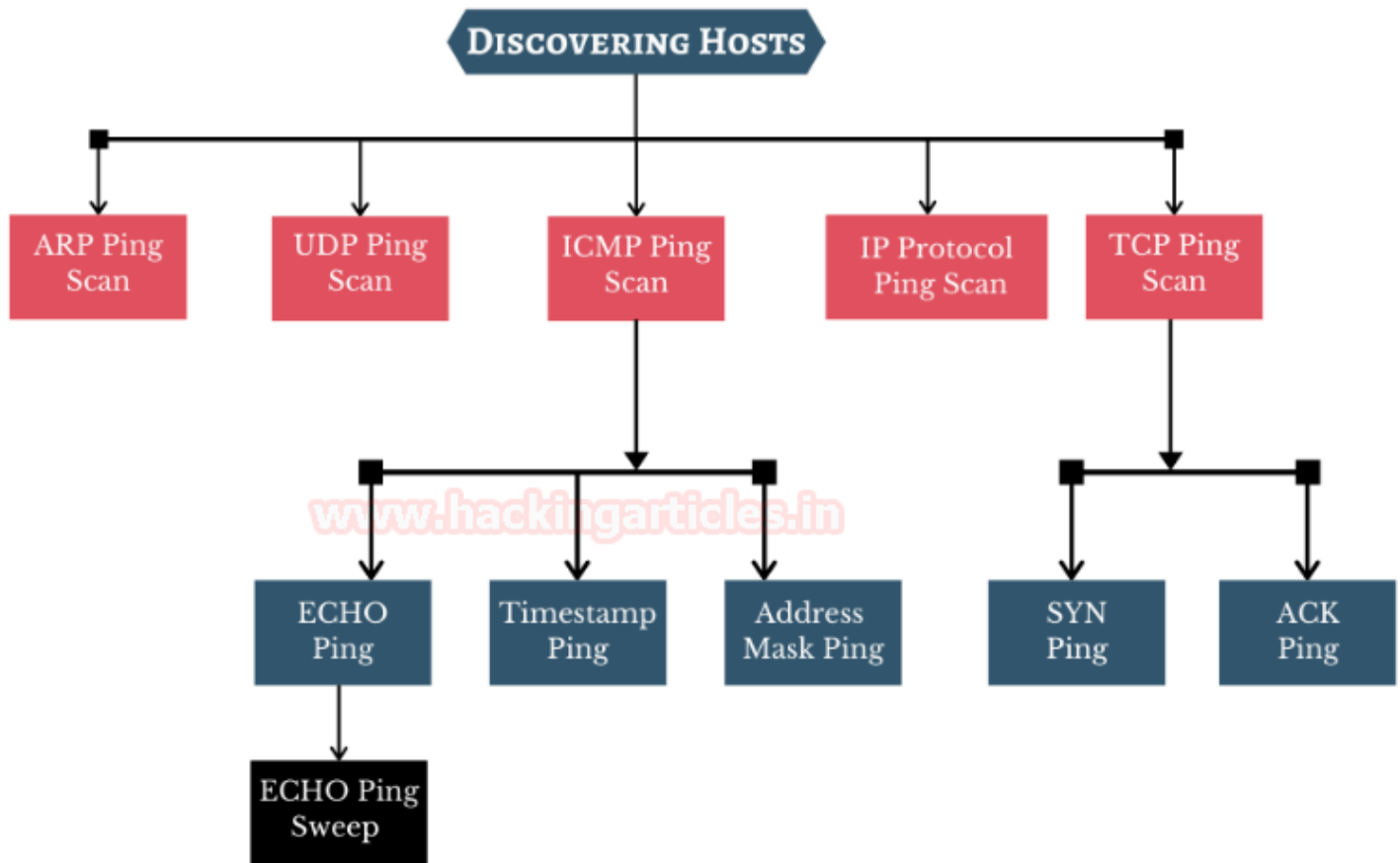
TCP Flags

First, let's get to know the basics about the communication Flags in TCP. The TCP header mainly consists of six flags which manage the connection between the systems and provide instructions to them. Each flag is of 1 bit and hence the size of TCP Flags is 6 bits. Now let us briefly understand each flag.

| FLAG | DESCRIPTION |
|------|---|
| SYN | It stands for Synchronize. It assists in notifying when a new sequence number is transmitted. The SYN flag usually represents the Three-Way Handshake. |
| ACK | It stands for Acknowledgement. It notifies the status of transmission of packets and also assists in identifying the what sequence number to expect next. |
| RST | It stands for Reset. This flag shows when there is any error in that connection and sets the flag to 1 and the connection is broken. |
| URG | It stands for Urgent. This flag usually commands to process the packets as soon as possible. |
| FIN | It stands for Finish. This flag is set as 1 to indicate no further transmission of packets. |
| PSH | It stands for Push. It is used to start and end data transfer and prevent occurrence of buffer deadlocks. |

Types of Scans

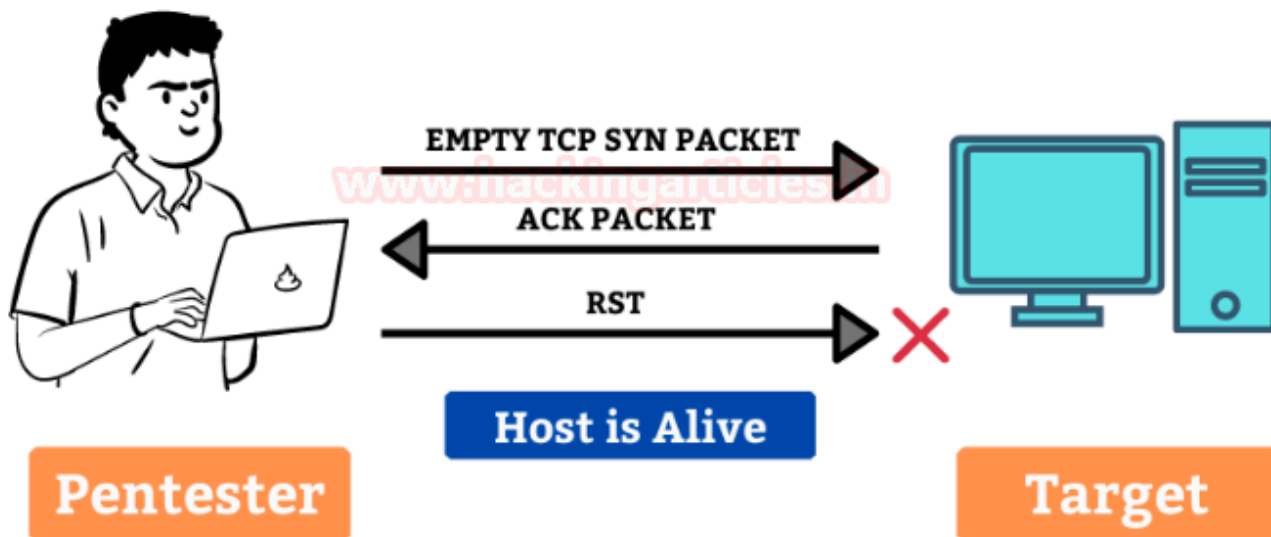
To discover the hosts in the network, various ping scan methods can be used.



TCP SYN Ping Scan

It is a method of host discovery which helps in looking for discovering if the ports are open and to also make sure if it matches the rules of the firewall. The Pentester can hence, send an empty SYN flag to the target to check where it is alive. Multiple ports can be defined in this scan type.

TCP SYN PING SCAN



The `-sP` command in Nmap only allows discovering online hosts. Whereas SYN Ping (`-PS`) sends a TCP SYN packet to the ports and if it is closed, the host responds with an RST packet. And if the ports requested are open there will be the response of TCP SYN/ACK and there will be a reset packet which will be sent to reset the connection.

```
nmap -sn -PS 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PS 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 07:13 EST
Nmap scan report for 192.168.1.108
Host is up (0.00030s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.12 seconds
```

The packets captured using Wireshark can be overserved

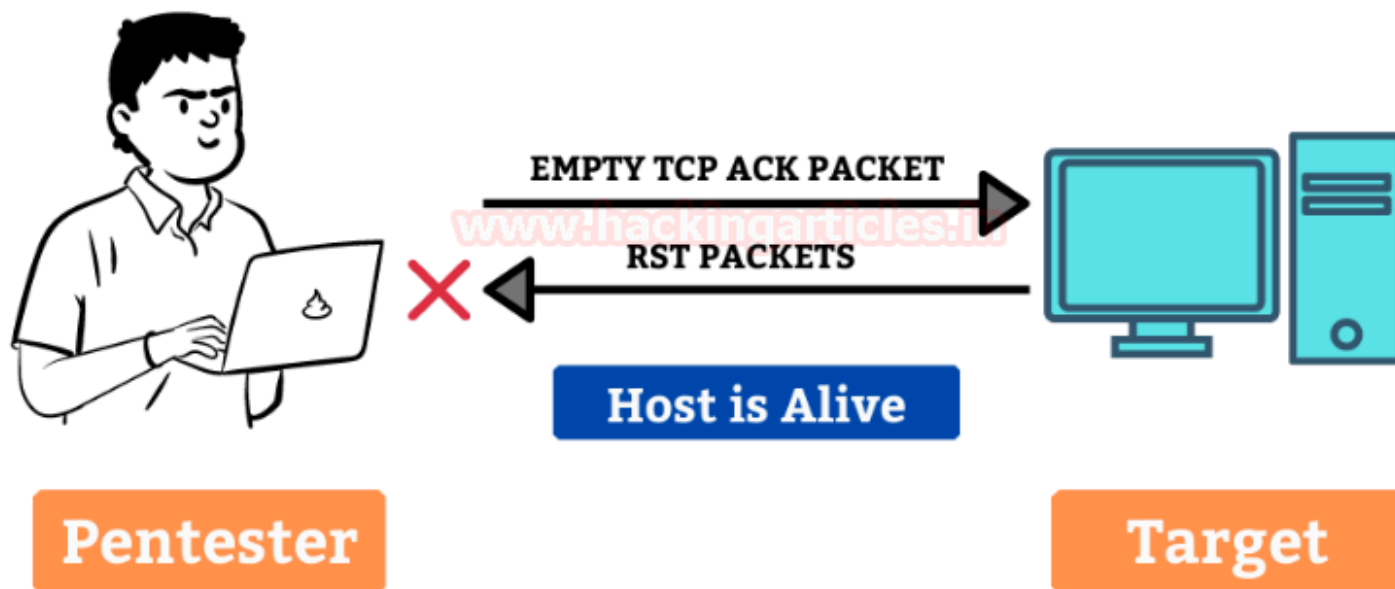
| No | Tin | Source | Destination | Protocol | Length | Info |
|-----|-----|---------------|---------------|----------|--------|--|
| ... | ... | 192.168.1.9 | 192.168.1.108 | TCP | 58 | 47752 → 80 [SYN] Seq=0 Win=1024 Len=0 |
| ... | ... | 192.168.1.108 | 192.168.1.9 | TCP | 60 | 80 → 47752 [SYN, ACK] Seq=0 Ack=1 Win= |
| ... | ... | 192.168.1.9 | 192.168.1.108 | TCP | 54 | 47752 → 80 [RST] Seq=1 Win=0 Len=0 |

The advantage of TCP SYN Ping scan is that the pentester can get the active/inactive status of the host without even creating a connection and hence it does not even create a log in the system or the network.

TCP ACK Ping Scan

It is a method of host discovery which is similar to TCP SYN Ping scan but slightly differs. This scan also makes use of Port 80. The pentester sends an empty TCP packet to the target and as there is no connection between them, it will receive an Acknowledgement packet and will then reset and terminate the request

TCP ACK PING SCAN



This command is used to determine the target's response and also check if the SYN packets or ICMP echo requests are blocked as of in the latest firewalls

```
nmap -sn -PA 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PA 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 07:14 EST
Nmap scan report for 192.168.1.108
Host is up (0.00023s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.12 seconds
root@kali:~#
```

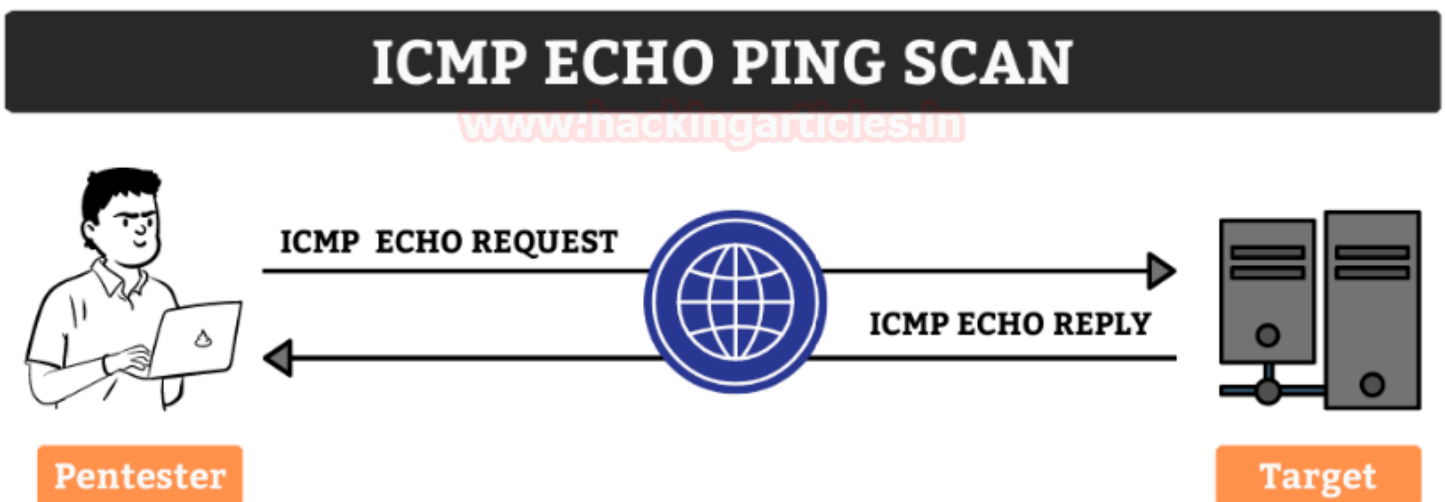
The Packets captured in the Wireshark can be observed here.

| No | Tin | Source | Destination | Protocol | Length | Info |
|-----|-----|---------------|---------------|----------|--------|------------------------------|
| ... | ... | 192.168.1.9 | 192.168.1.108 | TCP | 54 | 47131 → 80 [ACK] Seq=1 Ack=1 |
| ... | ... | 192.168.1.108 | 192.168.1.9 | TCP | 60 | 80 → 47131 [RST] Seq=1 Win=0 |

Some firewalls are configured to block on SYN ping packets, hence, in this case, this scan would be effective to bypass the firewall easily.

ICMP Echo Ping Scan

The ICMP Ping scan can be used to gather information about the target systems which makes it different from port scanning. The pentester can send an ICMP ECHO request to the target and getting an ICMP Echo reply in return.



ICMP is now ineffective on remote ICMP packets which have been blocked by admins. It can still be used to monitor local networks.

```
nmap -sn -PE 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PE 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 07:15 EST
Nmap scan report for 192.168.1.108
Host is up (0.00039s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds
root@kali:~#
```

The packets captured in the Wireshark can be observed.

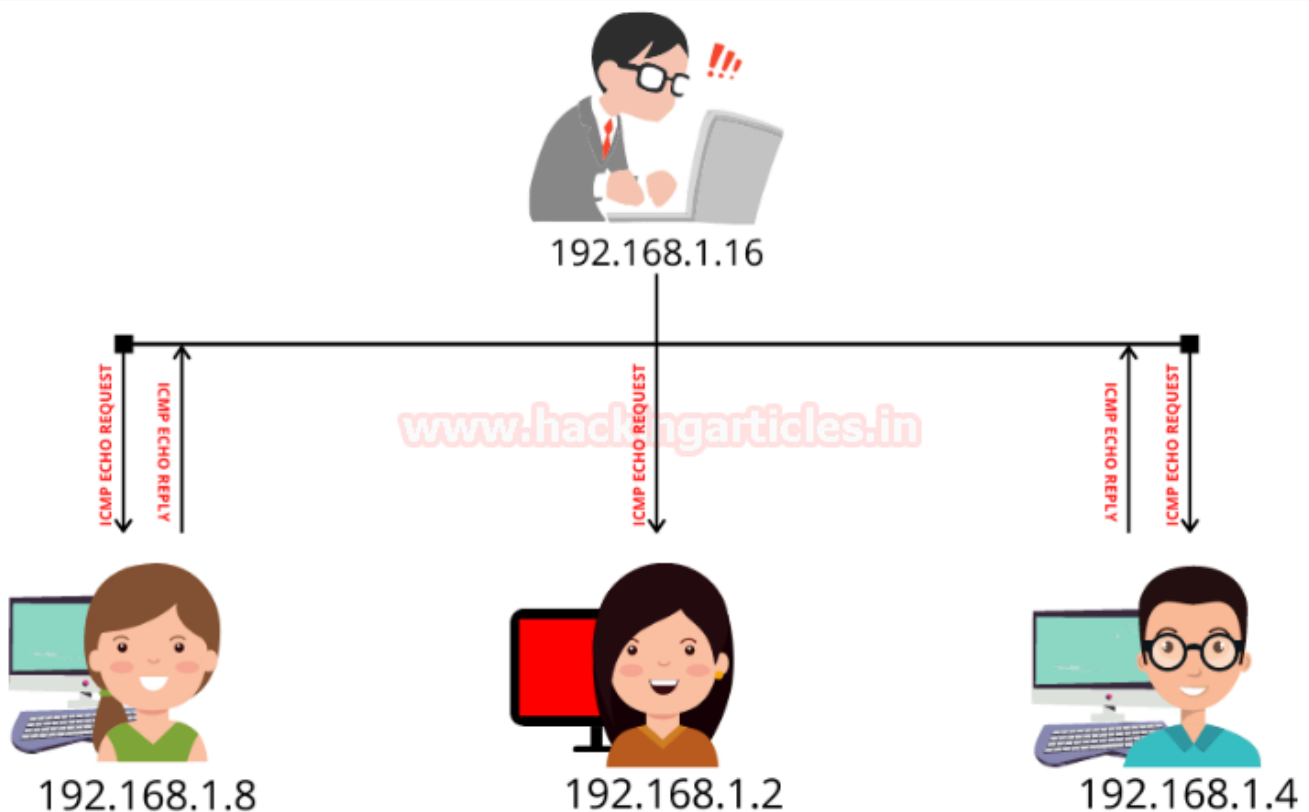
| File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help | | | | | | |
|--|-----|---------------|---------------|----------|--------|------------------------------|
| ip.addr == 192.168.1.108 | | | | | | |
| No | Tin | Source | Destination | Protocol | Length | Info |
| ... | ... | 192.168.1.9 | 192.168.1.108 | ICMP | 42 | Echo (ping) request id=0xdb6 |
| ... | ... | 192.168.1.108 | 192.168.1.9 | ICMP | 60 | Echo (ping) reply id=0xdb6 |

ICMP ECHO Ping Sweep

It is similar to Echo Ping Scan and is used to scan the active hosts from a given range of IP addresses. It sends ICMP requests to a huge number of targets and if a particular target is alive then it will return an ICMP reply.

```
nmap -sn -PE 192.168.1-10
```

ICMP ECHO PING SWEEP



ICMP Address Mask Scan

It is an older method of ICMP ECHO ping scanning. It gives out the information about the system and its subnet mask.

```
nmap -sn -PM 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PM 192.168.1.108
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 12:15 EST
Nmap scan report for 192.168.1.108
Host is up (0.00026s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
root@kali:~#
```

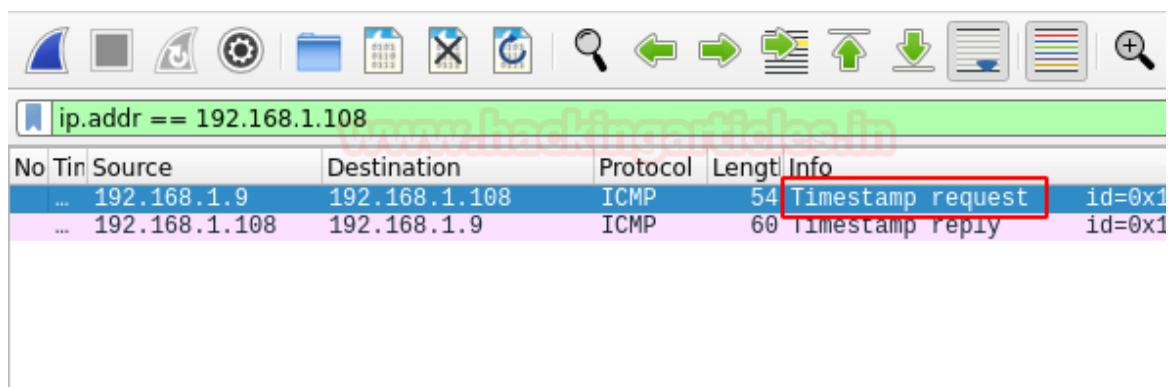
ICMP ECHO Timestamp scan

The pentester can adopt this technique in a particular condition when the system admin blocks the regular ICMP timestamp. It is usually used in synchronization of time.

```
nmap -sn -PP 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PP 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 07:17 EST
Nmap scan report for 192.168.1.108
Host is up (0.00059s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
root@kali:~#
```

The packets captured using Wireshark can be observed.



The image shows a Wireshark packet capture interface. The filter bar at the top is set to 'ip.addr == 192.168.1.108'. The packet list shows two ICMP packets. The first packet is a 'Timestamp request' from 192.168.1.9 to 192.168.1.108, with a length of 54 bytes. The second packet is a 'Timestamp reply' from 192.168.1.108 to 192.168.1.9, with a length of 60 bytes. Both packets have an ID of 0x1. The 'Timestamp request' packet is highlighted with a red box.

| No | Tin | Source | Destination | Protocol | Length | Info |
|-----|-----|---------------|---------------|----------|--------|--------------------------|
| ... | ... | 192.168.1.9 | 192.168.1.108 | ICMP | 54 | Timestamp request id=0x1 |
| ... | ... | 192.168.1.108 | 192.168.1.9 | ICMP | 60 | Timestamp reply id=0x1 |

UDP Ping Scan

The UDP Ping Scans uses a highly uncommon default port number 40125 to send packets to the target. It is similar to TCP Ping scan. The Pentester will send the UDP Packets to the target and if there is a response in return which means that the host is alive or else it is offline

UDP PING SCAN WHEN TARGET IS ACTIVE



www.hackingarticles.in

UDP PING SCAN WHEN TARGET IS INACTIVE



The advantage of UDP scan is that it can detect the systems which have firewalls with strict TCP rules and leaving UDP rules at ease.

```
nmap -sn -PU 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PU 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 12:06 EST
Nmap scan report for 192.168.1.108
Host is up (0.00032s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.12 seconds
root@kali:~#
```

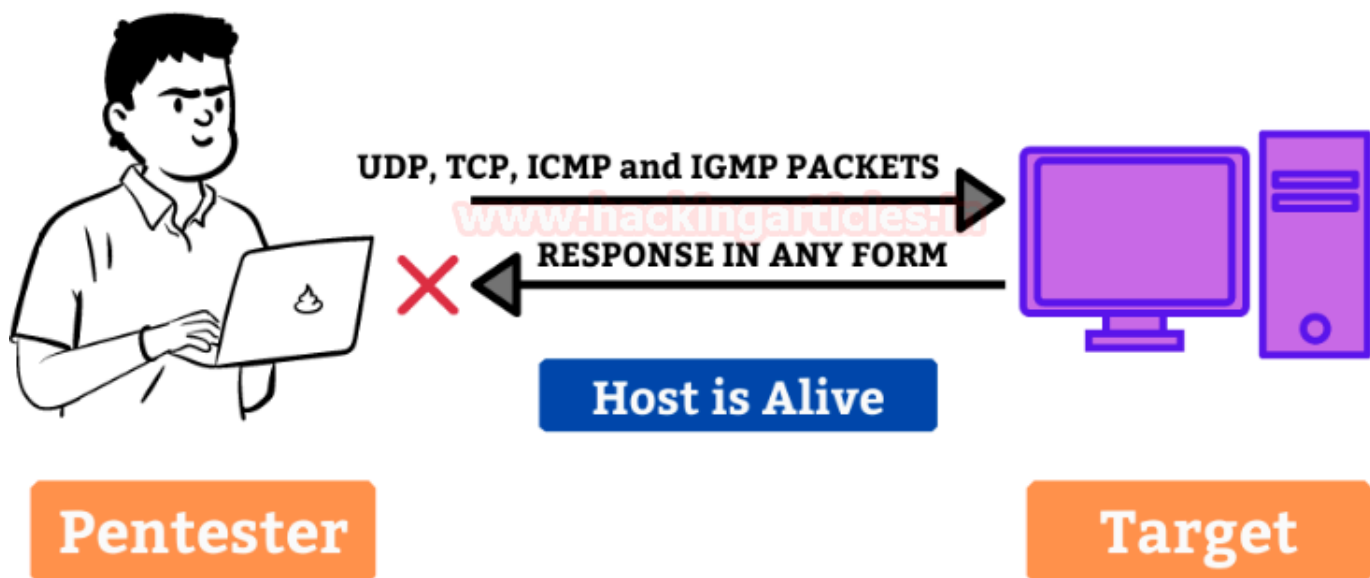
You can observe the packets sent using Wireshark.

| ip.addr == 192.168.1.108 | | | | | | |
|--------------------------|-----|-------------|---------------|----------|--------|----------------------|
| No | Tin | Source | Destination | Protocol | Length | Info |
| ... | | 192.168.1.9 | 192.168.1.108 | UDP | 82 | 63059 → 40125 Len=40 |

IP protocol ping scan

In this method, the pentester sends various packets using different IP protocols and hopes to get a response in return if the target is alive.

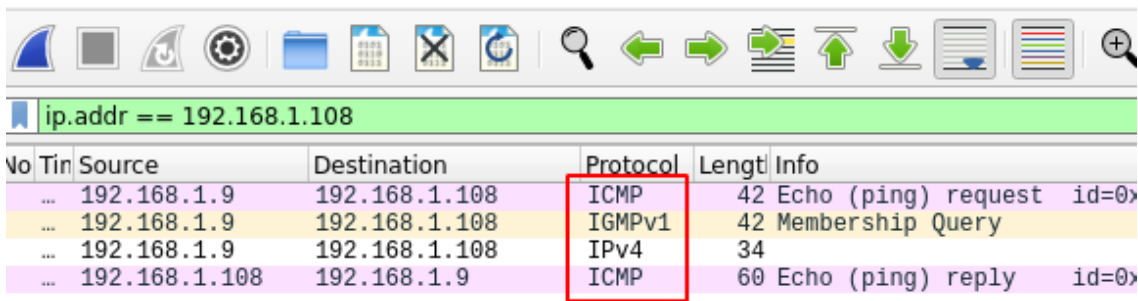
IP PROTOCOL PING SCAN



```
nmap -sn -PO 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PO 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 12:07 EST
Nmap scan report for 192.168.1.108
Host is up (0.00040s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
```

The packets captured can be observed using Wireshark.



| No | Time | Source | Destination | Protocol | Length | Info |
|-----|------|---------------|---------------|----------|--------|---------------------------|
| ... | ... | 192.168.1.9 | 192.168.1.108 | ICMP | 42 | Echo (ping) request id=0> |
| ... | ... | 192.168.1.9 | 192.168.1.108 | IGMPv1 | 42 | Membership Query |
| ... | ... | 192.168.1.9 | 192.168.1.108 | IPv4 | 34 | |
| ... | ... | 192.168.1.108 | 192.168.1.9 | ICMP | 60 | Echo (ping) reply id=0> |

No ping scan

In this method, host discovery is completely skipped. The pentester can use it to determine active machines for heavier scanning and to increase the speed of the network.

```
nmap -sn -PN 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PN 192.168.1.108 --disable-arp-ping
Host discovery disabled (-Pn). All addresses will be marked 'up' and s
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 12:10 EST
Nmap scan report for 192.168.1.108
Host is up.
Nmap done: 1 IP address (1 host up) scanned in 0.01 seconds
```

ARP ping scan

In this method, the ARP packets are sent to all the devices I the network although they are invisible due to the firewall. It is considered to be extremely efficient than other host discovery. It is mainly used for system discovery. It also mentions the latency.

ARP PING SCAN



```
nmap -sn -PR 192.168.1.108
```

```
root@kali:~# nmap -sn -PR 192.168.1.108
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 12:12 EST
Nmap scan report for 192.168.1.108
Host is up (0.00029s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.27 seconds
root@kali:~#
```

You can see the packets being captured in wireshark.

| No | Tin | Source | Destination | Protocol | Length | Info |
|-----|-----|-------------------|-------------------|----------|--------|-----------------------|
| ... | ... | VMware_b2:bb:77 | Broadcast | ARP | 42 | Who has 192.168.1.108 |
| ... | ... | VMware_c8:9c:50 | VMware_b2:bb:77 | ARP | 60 | 192.168.1.108 is at |
| ... | ... | TaicangT_69:a5:10 | VMware_b2:bb:77 | ARP | 60 | Who has 192.168.1.97 |
| ... | ... | VMware_b2:bb:77 | TaicangT_69:a5:10 | ARP | 42 | 192.168.1.9 is at 06 |
| ... | ... | TaicangT_69:a5:10 | VMware_c8:9c:50 | ARP | 60 | Who has 192.168.1.108 |
| ... | ... | VMware_c8:9c:50 | TaicangT_69:a5:10 | ARP | 60 | 192.168.1.108 is at |

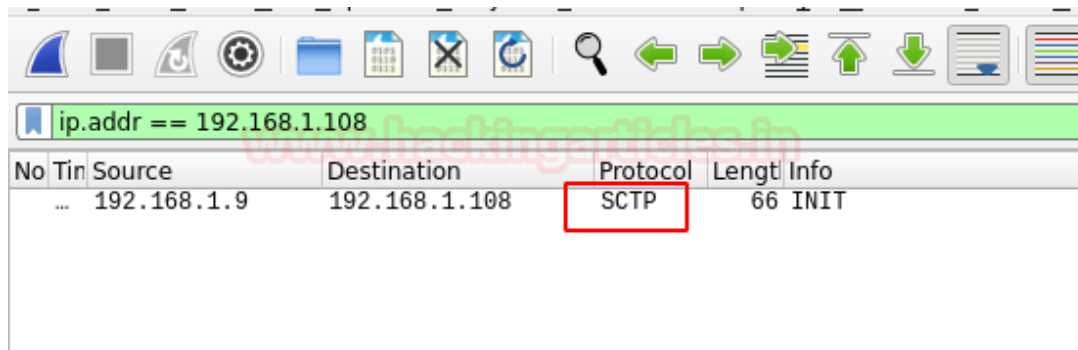
SCTP INIT Ping

It sends SCTP packet containing a minimal INIT chunk. Its default destination port is 80. The INIT chunk provides suggestion to the remote system that the pentester is attempting to establish an association.

```
nmap -sn -PY 192.168.1.108 --disable-arp-ping
```

```
root@kali:~# nmap -sn -PY 192.168.1.108 --disable-arp-ping
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 12:13 EST
Nmap scan report for 192.168.1.108
Host is up (0.00030s latency).
MAC Address: 00:0C:29:C8:9C:50 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
root@kali:~#
```

The packets that are captured can be observed.



The image shows the Wireshark interface with a packet capture filter set to 'ip.addr == 192.168.1.108'. A single packet is captured, and its details are shown in the bottom pane. The packet is of type SCTP, with a length of 66 bytes and information 'INIT'. The source IP is 192.168.1.9 and the destination IP is 192.168.1.108.

| No | Time | Source | Destination | Protocol | Length | Info |
|-----|------|-------------|---------------|----------|--------|------|
| ... | ... | 192.168.1.9 | 192.168.1.108 | SCTP | 66 | INIT |

Traceroute

Traceroutes are used after finishing scanning, by using the information from the scan results and to determine the port and protocol which will reach the target.

```
nmap -sn --traceroute 8.8.8.8
```

```
root@kali:~# nmap -sn --traceroute 8.8.8.8
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-20 11:38 EST
Nmap scan report for dns.google (8.8.8.8)
Host is up (0.0014s latency).

TRACEROUTE (using port 80/tcp)
HOP RTT ADDRESS
1 1.85 ms dsldevice.lan (192.168.1.1)
2 1.57 ms dns.google (8.8.8.8)
```

To get more information Traceroute you can refer to

[Working of Traceroute using Wireshark](#)

Reference: <https://nmap.org/book/man-host-discovery.html>