Nmap for Pentester: Port Status

February 20, 2018 By Raj Chandel

Hello friends, several times you might have used NMAP to performing Network scanning for enumerating active Port services of target machine but in some scenarios you don't get simple message if a port open or close.

Let's Begin

Requirement

- **Attacker's IP:** 192.168.1.109 [Kali Linux]
- **Target's IP:** 192.168.1.119 [Ubuntu]

The states of ports are not their essential properties; it depicts how nmap sees them. In nmap a port is divided into six states:

1. **Open:** This state means that an application on the target machine is listening for connections/packets on that port.
2. **Closed:** This state means ports have no application listening on them, though they could open up at any time.
3. **Filtered:** This state means that a firewall, filter, or other network obstacle is blocking the port so that Nmap cannot tell whether it is open or closed.
4. **Unfiltered:** ports are classified as unfiltered when they are responsive to Nmap's probes, but Nmap cannot determine whether they are open or closed.
5. **Open/Filtered:** This indicates that the port was filtered or open but Nmap couldn't establish the state.
6. **Closed/Filtered**: This indicates that the port was filtered or closed but Nmap couldn't establish the state.

Open Port

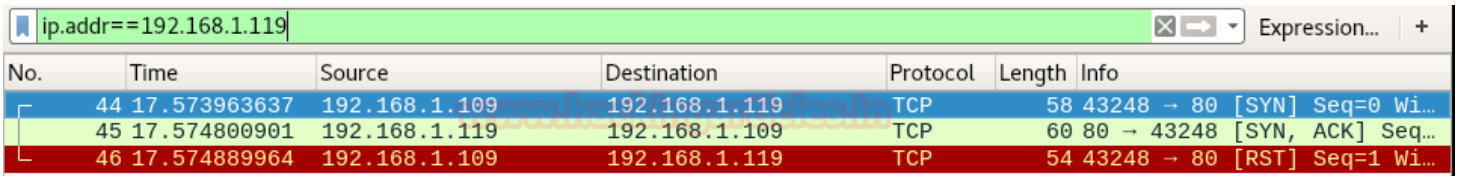
In this case a service or application running on a port is actively accepting TCP, UDP connections. We send TCP packets to port 80 of target machine. We find that the port is open.

```
nmap -p80 192.168.1.119
```

```
root@kali:~# nmap -p80 192.168.1.119

Starting Nmap 7.60 ( https://nmap.org ) at 2018-02-17 04:29 EST
Nmap scan report for 192.168.1.119
Host is up (0.00043s latency).

PORT   STATE SERVICE
80/tcp open  http
MAC Address: 00:0C:29:48:22:FD (VMware)

Nmap done: 1 IP address (1 host up) scanned in 13.25 seconds
root@kali:~#
```

We take a look at wireshark and find that 3 way-handshake occurs as given below.

- Nmap sends SYN packet on port 80
- Nmap received SYN, ACK packet as response from port 80 which denotes port 80 is open.
- Nmap sends RST packet



Closed Port

In this case a service or application on a port is accessible but no application is running on it. When a port is in closed state it sends RST with ACK packet when it receives TCP SYN packet

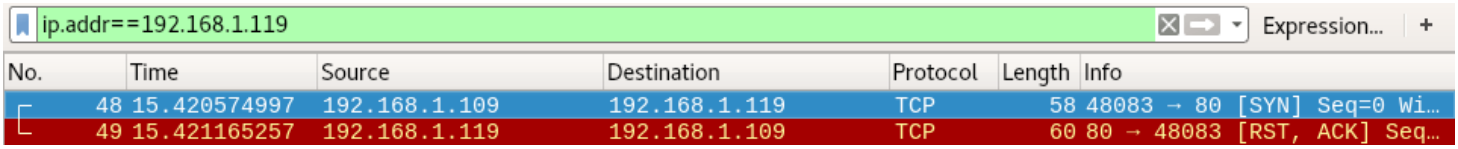
```
nmap -p80 192.168.1.119
```

Now we have used SYN scan to send TCP SYN packets on port 80 of target machine and found that the target is **closed**. That is because as soon as it receives TCP SYN packet it sends back TCP RST, ACK packet.

```
root@kali:~# nmap -p80 192.168.1.119

Starting Nmap 7.60 ( https://nmap.org ) at 2018-02-17 04:30 EST
Nmap scan report for 192.168.1.119
Host is up (0.00087s latency).

PORT   STATE  SERVICE
80/tcp closed http
MAC Address: 00:0C:29:48:22:FD (VMware)

Nmap done: 1 IP address (1 host up) scanned in 13.29 seconds
root@kali:~#
```

We will check wireshark to find more information, as expected as soon as the target machine received TCP SYN packet it replied with TCP RST and NMAP interpreted it as port is closed.

- Nmap sends SYN packet on port 80
- Nmap received RST, ACK packet as response from port 80 which denotes port 80 is closed.



Filtered Port

In this case Nmap is unable to determine whether a port is open because packet filtering is preventing the packets from reaching the port. When a packet is dropped Nmap retries several times just in case the probe was dropped due to network congestion rather than filtering. This slows down the scan dramatically.

Let's use iptables to drop TCP packets on the target machine.

```
iptables -I INPUT -p tcp -j DROP
```

```
root@ubuntu:~# iptables -I INPUT -p tcp -j DROP
root@ubuntu:~#
```

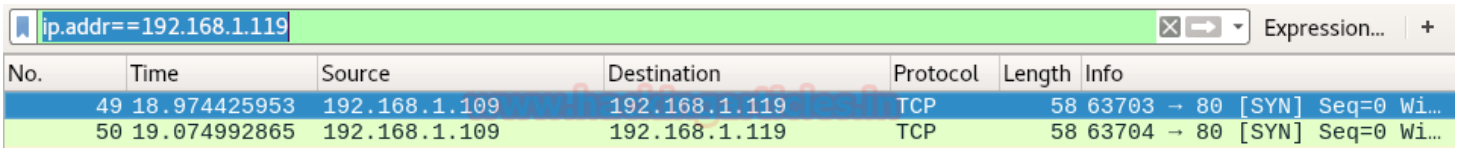
Now when we scan the target machine, the packets will be dropped as soon as it receives TCP packets.

```
nmap -p80 192.168.1.119
```

From given below image you can observe that it is now showing state "filtered" for port 80.

```
root@kali:~# nmap -p80 192.168.1.119

Starting Nmap 7.60 ( https://nmap.org ) at 2018-02-17 04:33 EST
Nmap scan report for 192.168.1.119
Host is up (0.00081s latency).

PORT   STATE    SERVICE
80/tcp filtered http
MAC Address: 00:0C:29:48:22:FD (VMware)

Nmap done: 1 IP address (1 host up) scanned in 13.45 seconds
root@kali:~#
```

Let's take a look at wireshark we find that when Nmap send TCP SYN packet we get no reply from the target machine. This means that a packet filter or firewall is dropping our packets.



Unfiltered Port

The unfiltered state means that a port is accessible, but Nmap is unable to determine whether it is open or closed. Only the ACK scan, which is used to map firewall rulesets, classifies ports into this state. Scanning unfiltered ports with other scan types such as Window scan, SYN scan, or FIN scan, may help resolve whether the port is open.

We use iptables to drop any TCP packet coming to port 80 in target machine.

```
iptables -I INPUT -p tcp --dport=80 -j DROP
```

```
root@ubuntu:~# iptables -I INPUT -p tcp --dport=80 -j DROP
root@ubuntu:~#
```

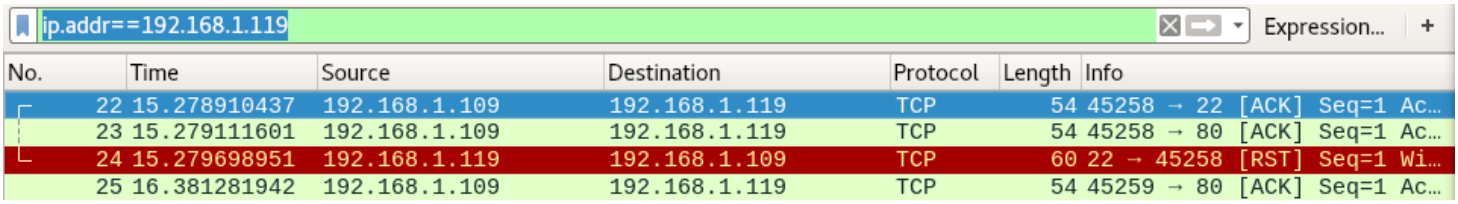
Now we use **nmap ACK scan** to scan the target machine to check if there is any firewall or not.

```
nmap -sA -p22,80 192.168.1.119
```

As we can see in given below image the port without firewall shows unfiltered as Nmap is unable to determine if it is open or close.

```
root@kali:~# nmap -sA -p22,80 192.168.1.119

Starting Nmap 7.60 ( https://nmap.org ) at 2018-02-17 04:51 EST
Nmap scan report for 192.168.1.119
Host is up (0.0012s latency).

PORT   STATE      SERVICE
22/tcp unfiltered ssh
80/tcp filtered   http
MAC Address: 00:0C:29:48:22:FD (VMware)

Nmap done: 1 IP address (1 host up) scanned in 14.44 seconds
root@kali:~#
```

We can see in wireshark that for **port 22** we get a **RST packet** whereas in case of port 80 the packet is dropped by the target machine.



Open|Filtered Port

In this case nmap is unable to determine if a port is open or filtered. This occurs for scan types in which open ports give no response. The lack of response could also mean that a packet filter dropped the probe or any response it

elicited. So Nmap does not know for sure whether the port is open or being filtered. The UDP, IP protocol, FIN, NULL, and Xmas scans classify ports this way.

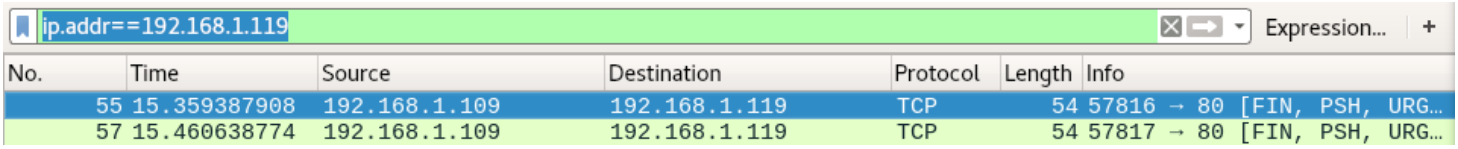
Let's use nmap Xmas scan to scan the target machine.

```
nmap -sX -p80 192.168.1.119
```

As we can see the nmap scan shows us the port to be open | filtered.

```
root@kali:~# nmap -sX -p80 192.168.1.119

Starting Nmap 7.60 ( https://nmap.org ) at 2018-02-17 04:36 EST
Nmap scan report for 192.168.1.119
Host is up (0.00032s latency).

PORT    STATE         SERVICE
80/tcp open|filtered http
MAC Address: 00:0C:29:48:22:FD (VMware)

Nmap done: 1 IP address (1 host up) scanned in 13.47 seconds
```


We will check wireshark to analyze the packets sent by nmap and we can see we don't get a reply even if the port is open.



Closed|Filtered Port

This state is used when Nmap is unable to determine whether a port is closed or filtered. It is only used for the IP ID **idle scan**.

We use iptables on our target machine to drop incoming TCP packets on the target machine.

```
iptables -I INPUT -p tcp -j DROP
```

```
root@ubuntu:~# iptables -I INPUT -p tcp -j DROP
root@ubuntu:~#
```


We will do an IP ID idle scan on the target machine using 192.168.1.107 as our zombie.

```
nmap -p80 -sI 192.168.1.107 192.168.1.119
```

As we can see in idle scan the zombie it is showing state closed|filtered for port 80.

```
root@kali:~# nmap -p80 -sI 192.168.1.107 192.168.1.119
WARNING: Many people use -Pn w/Idlescan to prevent pings from their true IP.  On the other hand, timing
info Nmap gains from pings can allow for faster, more reliable scans.

Starting Nmap 7.60 ( https://nmap.org ) at 2018-02-17 04:54 EST
Idle scan using zombie 192.168.1.107 (192.168.1.107:443); Class: Incremental
Nmap scan report for 192.168.1.119
Host is up (0.075s latency).

PORT   STATE          SERVICE
80/tcp closed|filtered http
MAC Address: 00:0C:29:48:22:FD (VMware)

Nmap done: 1 IP address (1 host up) scanned in 25.80 seconds
root@kali:~#
```

An idle scan consists of three steps that are repeated for each port:

1. Probe the zombie's IP ID and record it.
2. Forge a SYN packetfrom the zombie and send it to the desired port on the target. Depending on the port state, the target's reaction may or may not cause the zombie's IP ID to be incremented.
3. Probe the zombie's IP ID again. The target port state is then determined by comparing this new IP ID with the one recorded in step 1.

We check Wireshark and find that find the entire process.

79	25.894979100	192.168.1.119	192.168.1.107	TCP	58 44029 → 443 [SYN, ACK] Se…
82	25.923506000	192.168.1.107	192.168.1.119	TCP	60 443 → 44029 [RST] Seq=1 W…
83	25.946400216	192.168.1.119	192.168.1.107	TCP	58 [TCP Port numbers reused]…
84	25.948774628	192.168.1.107	192.168.1.119	TCP	60 443 → 44029 [RST] Seq=1 W…
85	25.998508416	192.168.1.119	192.168.1.107	TCP	58 [TCP Port numbers reused]…
86	26.000205686	192.168.1.107	192.168.1.119	TCP	60 443 → 44029 [RST] Seq=1 W…
87	26.050172982	192.168.1.119	192.168.1.107	TCP	58 [TCP Port numbers reused]…
88	26.053268538	192.168.1.107	192.168.1.119	TCP	60 443 → 44029 [RST] Seq=1 W…
91	26.352840661	192.168.1.107	192.168.1.119	TCP	58 443 → 80 [SYN] Seq=0 Win=…
100	26.799632443	192.168.1.107	192.168.1.119	TCP	58 [TCP Retransmission] 443 …

Source: https://nmap.org/book/man.html