Nmap Cheatsheet

Contents

This document shall contain Quick-and-Dirty notes about nmap. I must keep it up-to-date because I feel a bit inundated in this cybersecurity journey to be frank.

1 Basic Nmap Commands

1.1 Service and Version Detection (-sV)

This is the Service Detection flag *(yes; `-sV` is a single flag, not a combination of both `s` AND `V`)*, which will tell you the name and description of the identified services.

Service and Version Detection

```
$ sudo nmap -sV {target_IP}

Starting Nmap 7.80 ( https://nmap.org ) at 2025-03-28 06:27 GMT
Nmap scan report for 10.10.171.202
Host is up (0.00011s latency).
Not shown: 994 closed ports
PORT     STATE SERVICE    VERSION
7/tcp    open  echo
9/tcp    open  tcpwrapped
13/tcp   open  daytime?
17/tcp   open  qotd?
22/tcp   open  ssh        OpenSSH 9.6p1 Ubuntu 3ubuntu13.5 (Ubuntu
    Linux; protocol 2.0)
8008/tcp open  http       lighttpd 1.4.74

2 services unrecognized despite returning data. If you know the
   service/version, please submit the following fingerprints at
   https://nmap.org/cgi-bin/submit.cgi?new-service :
# [Detailed fingerprint data omitted for brevity]

MAC Address: 02:89:EC:B7:74:EF (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results
   at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.00 seconds
```

1.2 Default Script Scan (-sC)

I sometimes use this one instead of `-sV` because it runs *default scripts*[1], which can give out additional information depending on the services running on the target. You can see a comparison in outputs between the two flags in the two boxes below.

Default Script Scan

```
$ sudo nmap -p- -sC {target_IP} # Do you notice how I had to scan all ports, not
just the top 1000 most common?

Starting Nmap 7.80 ( https://nmap.org ) at 2025-03-28 09:22 GMT
Nmap scan report for 10.10.219.233
Host is up (0.00024s latency).
Not shown: 65526 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
25/tcp  open  smtp
|_smtp-commands: debra2.thm.local, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS
    , ENHANCEDSTATUSCODES, 8BITMIME, DSN, CHUNKING,
| ssl-cert: Subject: commonName=debra2.thm.local
| Not valid before: 2021-08-10T12:10:58
|_Not valid after:  2031-08-08T12:10:58
|_ssl-date: TLS randomness does not represent time
53/tcp  open  domain
| dns-nsid:
|_  bind.version: 9.18.28-1~deb12u2-Debian
80/tcp  open  http
|_http-title: Welcome to nginx on Debian!
110/tcp open  pop3
|_pop3-capabilities: PIPELINING SASL UIDL STLS AUTH-RESP-CODE RESP-CODES CAPA TOP
| ssl-cert: Subject: commonName=debra2.thm.local
| Not valid before: 2021-08-10T12:10:58
|_Not valid after:  2031-08-08T12:10:58
111/tcp open  rpcbind
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  3,4          111/tcp6  rpcbind
|_  100000  3,4          111/udp6  rpcbind
143/tcp open  imap
|_imap-capabilities: more LOGIN-REFERRALS have IDLE post-login STARTTLS listed
    ENABLE capabilities LOGINDISABLEDA0001 Pre-login SASL-IR OK ID LITERAL+
    IMAP4rev1
| ssl-cert: Subject: commonName=debra2.thm.local
| Not valid before: 2021-08-10T12:10:58
|_Not valid after:  2031-08-08T12:10:58
993/tcp open  imaps
|_imap-capabilities: LOGIN-REFERRALS more IDLE capabilities OK post-login ENABLE
    listed have Pre-login SASL-IR AUTH=PLAINA0001 ID LITERAL+ IMAP4rev1
| ssl-cert: Subject: commonName=debra2.thm.local
| Not valid before: 2021-08-10T12:10:58
|_Not valid after:  2031-08-08T12:10:58
995/tcp open  pop3s
|_pop3-capabilities: PIPELINING SASL(PLAIN) UIDL USER AUTH-RESP-CODE RESP-CODES
    CAPA TOP
| ssl-cert: Subject: commonName=debra2.thm.local
| Not valid before: 2021-08-10T12:10:58
|_Not valid after:  2031-08-08T12:10:58
MAC Address: 02:DD:7B:88:3D:75 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 20.77 seconds
```

[1] Default NSE Scripts, Nmap.org

Service Version Detection for Comparison

```
$ sudo nmap -sV {target_IP}

Starting Nmap 7.80 ( https://nmap.org ) at 2025-03-28 09:21 GMT
Nmap scan report for 10.10.219.233
Host is up (0.0062s latency).
Not shown: 991 closed ports
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 9.2p1 Debian 2+deb12u3 (protocol
   2.0)
25/tcp  open  smtp     Postfix smtpd
53/tcp  open  domain   ISC BIND 9.18.28-1~deb12u2 (Debian Linux)
80/tcp  open  http     nginx 1.22.1
110/tcp open  pop3     Dovecot pop3d
111/tcp open  rpcbind  2-4 (RPC #100000)
143/tcp open  imap     Dovecot imapd
993/tcp open  ssl/imap Dovecot imapd
995/tcp open  ssl/pop3 Dovecot pop3d
MAC Address: 02:DD:7B:88:3D:75 (Unknown)
Service Info: Host:  debra2.thm.local; OS: Linux; CPE: cpe:/o:
   linux:linux_kernel

Service detection performed. Please report any incorrect results
   at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.00 seconds
```

1.3 OS Detection (-O)

Nmap sends a series of TCP and UDP packets to the remote host and examines practically every bit in the responses.

OS Detection Scan

```
$ sudo nmap -O {target_IP}

Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-28 18:41 +03
Nmap scan report for # {redacted}
Host is up (0.0044s latency).
Not shown: 994 closed tcp ports (reset)
PORT     STATE    SERVICE
21/tcp   filtered ftp
22/tcp   filtered ssh
23/tcp   filtered telnet
53/tcp   open     domain
80/tcp   open     http
8022/tcp filtered oa-system
MAC Address: # {redacted}
Device type: general purpose
Running: Linux 3.X|4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
   cpe:/o:linux:linux_kernel:5
OS details: Linux 3.10 - 4.11, Linux 5.10 - 5.13
Network Distance: 1 hop
```

1.4 Aggressive Scan

What if you can have both OS detection and version detection with a single scan? That's what the aggressive scan is for. It enables OS detection, version detection, script scanning, and traceroute.

Aggressive Scan

```
$ sudo nmap -A {target_IP}

Starting Nmap 7.80 ( https://nmap.org ) at 2025-03-28 09:21 GMT
Nmap scan report for 10.10.219.233
Host is up (0.0062s latency).
Not shown: 991 closed ports
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 9.2p1 Debian 2+deb12u3 (protocol 2.0)
| ssh-hostkey:
|   256 b9:bc:8f:01:5f:59:23:d3:3a:a2:2d:04:10:e5:04:2d (ECDSA)
|_  256 c0:11:12:52:14:b3:e2:3d:41:bc:3e:94:bb:73:5f:89 (ED25519)
25/tcp  open  smtp     Postfix smtpd
|_smtp-commands: debra2.thm.local, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN,
   CHUNKING,
| ssl-cert: Subject: commonName=debra2.thm.local
| Not valid before: 2021-08-10T12:10:58
|_Not valid after:  2031-08-08T12:10:58
53/tcp  open  domain   ISC BIND 9.18.28-1~deb12u2 (Debian Linux)
| dns-nsid:
|_  bind.version: 9.18.28-1~deb12u2-Debian
80/tcp  open  http     nginx 1.22.1
|_http-title: Welcome to nginx on Debian!
|_http-server-header: nginx/1.22.1
110/tcp open  pop3     Dovecot pop3d
|_pop3-capabilities: PIPELINING SASL UIDL STLS AUTH-RESP-CODE RESP-CODES CAPA TOP
| ssl-cert: Subject: commonName=debra2.thm.local
| Not valid before: 2021-08-10T12:10:58
|_Not valid after:  2031-08-08T12:10:58
111/tcp open  rpcbind  2-4 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  3,4          111/tcp6  rpcbind
|_  100000  3,4          111/udp6  rpcbind
143/tcp open  imap     Dovecot imapd
|_imap-capabilities: more LOGIN-REFERRALS have IDLE post-login STARTTLS listed ENABLE capabilities LOGINDISABLEDA0001 Pre-
   login SASL-IR OK ID LITERAL+ IMAP4rev1
| ssl-cert: Subject: commonName=debra2.thm.local
| Not valid before: 2021-08-10T12:10:58
|_Not valid after:  2031-08-08T12:10:58
993/tcp open  imaps?
| fingerprint-strings:
|   DNSStatusRequestTCP, DNSVersionBindReqTCP, Help, RPCCheck, SSLSessionReq, TerminalServerCookie:
|     * OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE LITERAL+ AUTH=PLAIN] Dovecot ready.
|   GenericLines, NULL:
|     * OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE LITERAL+ AUTH=PLAIN] Dovecot (Debian) ready.
|   GetRequest:
|     * OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE LITERAL+ AUTH=PLAIN] Dovecot ready.
|     BAD Command received in invalid state.
|   HTTPOptions:
|     * OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE LITERAL+ AUTH=PLAIN] Dovecot ready.
|     BAD Command received in invalid state.
|   RTSPRequest:
|     * OK [CAPABILITY IMAP4rev1 SASL-IR LOGIN-REFERRALS ID ENABLE IDLE LITERAL+ AUTH=PLAIN] Dovecot ready.
|     BAD Command received in invalid state.
|_    OPTIONS RTSP/1.0
| ssl-cert: Subject: commonName=debra2.thm.local
| Not valid before: 2021-08-10T12:10:58
|_Not valid after:  2031-08-08T12:10:58
995/tcp open  pop3s?
| fingerprint-strings:
|   DNSStatusRequestTCP, DNSVersionBindReqTCP, GenericLines, GetRequest, HTTPOptions, Help, NULL, RPCCheck, RTSPRequest,
   SSLSessionReq, TerminalServerCookie:
|_    +OK Dovecot (Debian) ready.
| ssl-cert: Subject: commonName=debra2.thm.local
| Not valid before: 2021-08-10T12:10:58
|_Not valid after:  2031-08-08T12:10:58
MAC Address: 02:DD:7B:88:3D:75 (Unknown)
Device type: general purpose
Running: Linux 5.X
OS CPE: cpe:/o:linux:linux_kernel:5
OS details: Linux 5.0 - 5.4
Network Distance: 1 hop
Service Info: Host:  debra2.thm.local; OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   6.16 ms 10.10.219.233

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 22.77 seconds
```

2 Host Discovery

2.1 ARP Scan (-PR)

The ARP scan, `-PR`, is what you'd typically use if you're already in the network that you want to scan for live systems. The `-sn` flag here is necessary because it prevents nmap from scanning for open ports after the ARP scan.

ARP Scan

```
$ sudo nmap -sn -PR 192.168.100.7/24

Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-29 02:07 +03
Nmap scan report for 192.168.100.1
Host is up (0.0016s latency).
MAC Address: # This will show for all the live hosts, for all the
upcoming Discovery Host scans in this document, but I'll redact
them just in case.
Nmap scan report for 192.168.100.9
Host is up (0.092s latency).
Nmap scan report for 192.168.100.12
Host is up (0.054s latency).
Nmap scan report for 192.168.100.13
Host is up (0.11s latency).
Nmap scan report for 192.168.100.15
Host is up (0.082s latency).
Nmap scan report for 192.168.100.16
Host is up (0.10s latency).
Nmap scan report for 192.168.100.25
Host is up (0.053s latency).
Nmap scan report for 192.168.100.28
Host is up (0.061s latency).
Nmap scan report for 192.168.100.7
Host is up.
Nmap done: 256 IP addresses (9 hosts up) scanned in 28.27 seconds
```

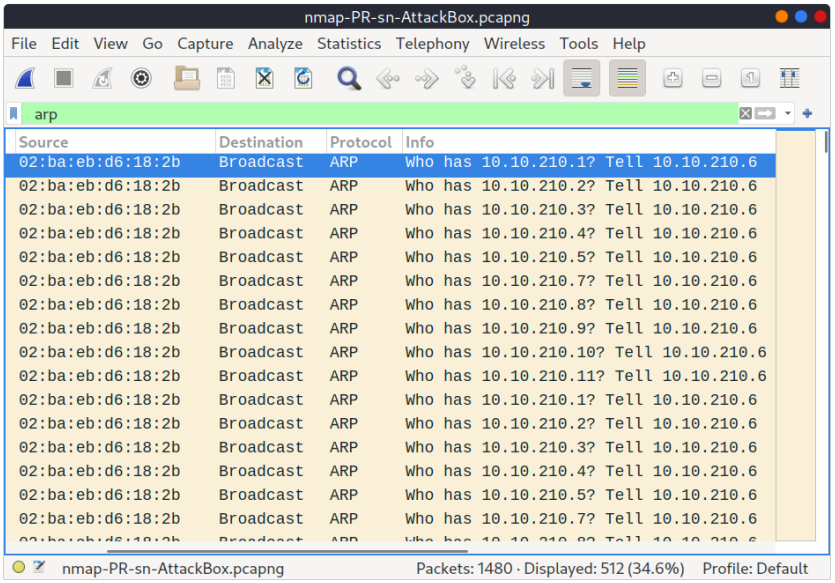


Figure 1: Wireshark capture of ARP scan packets, a network different from the aforementioned by the way.

2.2 ICMP Scans

We'd rarely send these ICMP packets for the discovery of our system's own subnet, ARP Scans would be better in that case since firewalls tend to block ICMP packets a lot.

2.2.I Echo Scan (-PE)

Either way, the `-PE` scan sends the ICMP Type 8 packets (i.e. same as your `ping` command).

ICMP Echo Scan (-PE)

```
$ sudo nmap -sn -PE 192.168.100.7/24

Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-29 03:06 +03
Nmap scan report for 192.168.100.1
Host is up (0.0039s latency).
Nmap scan report for 192.168.100.12
Host is up (0.33s latency).
Nmap scan report for 192.168.100.13
Host is up (0.74s latency).
Nmap scan report for 192.168.100.15
Host is up (0.072s latency).
Nmap scan report for 192.168.100.16
Host is up (0.32s latency).
Nmap scan report for 192.168.100.25
Host is up (0.11s latency).
Nmap scan report for 192.168.100.28
Host is up (0.73s latency).
Nmap scan report for 192.168.100.7
Host is up.
Nmap done: 256 IP addresses (8 hosts up) scanned in 9.18 seconds
```

Do you notice how it discovered 8 devices, whilst the previous ARP scan discovered 9?

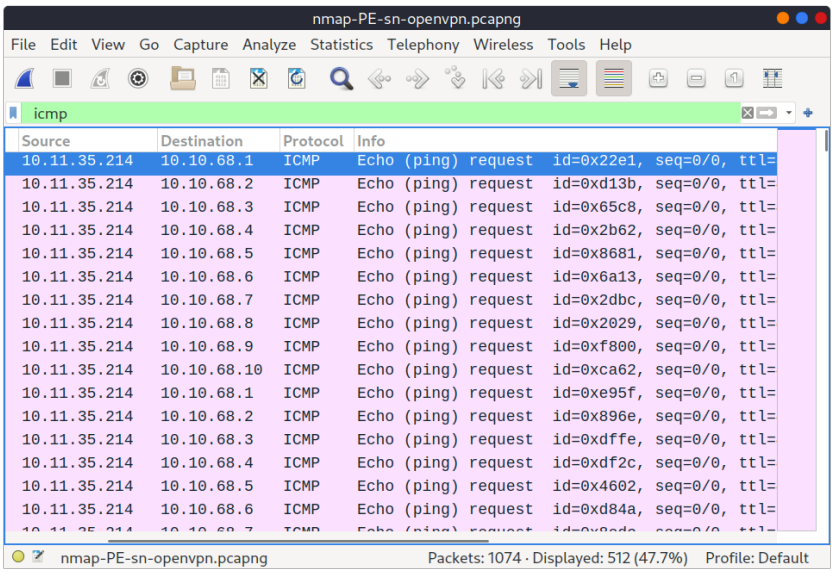


Figure 2: Wireshark capture of Echo Request packets, a network different from the aforementioned by the way.

2.2.II Timestamp Scan (-PP)

Because ICMP echo requests tend to be blocked, you might also consider ICMP Timestamp (ICMP Type 13) to tell if a system is online.

ICMP Timestamp Scan (-PP)

```
$ sudo nmap -sn -PP 192.168.100.7/24

Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-29 03:53 +03
Nmap scan report for 192.168.100.1
Host is up (0.0016s latency).
Nmap scan report for 192.168.100.9
Host is up (0.075s latency).
Nmap scan report for 192.168.100.12
Host is up (0.12s latency).
Nmap scan report for 192.168.100.13
Host is up (0.083s latency).
Nmap scan report for 192.168.100.14
Host is up (0.058s latency).
Nmap scan report for 192.168.100.15
Host is up (0.033s latency).
Nmap scan report for 192.168.100.16
Host is up (0.073s latency).
Nmap scan report for 192.168.100.25
Host is up (0.077s latency).
Nmap scan report for 192.168.100.7
Host is up.
Nmap done: 256 IP addresses (9 hosts up) scanned in 14.88 seconds
```

We got "9 hosts up" once again!

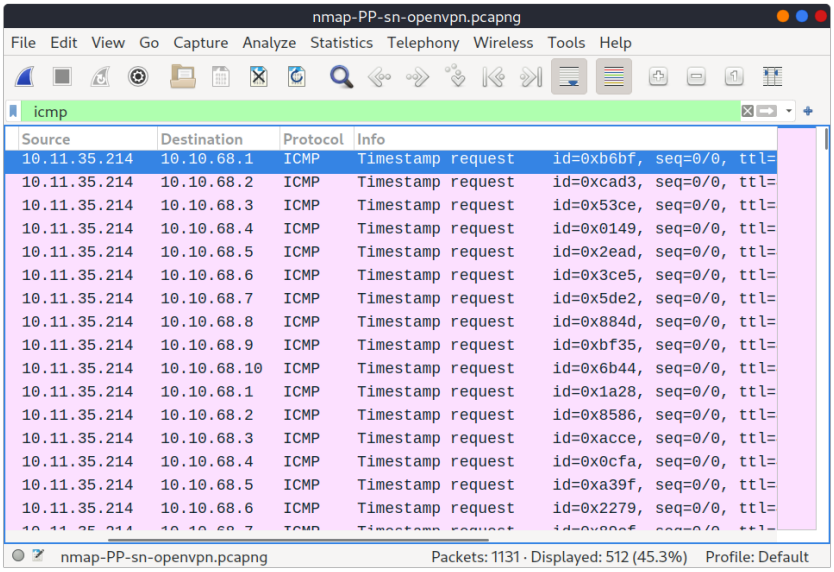


Figure 3: Wireshark capture of Timestamp Request packets.

2.2.III Address Mask Scan (-PM)

Similarly, Nmap uses address mask queries (ICMP Type 17) and checks whether it gets an address mask reply (ICMP Type 18).

ICMP Address Mask Scan (-PM)

```
$ sudo nmap -sn -PM 192.168.100.7/24

Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-29 03:53 +03
Nmap scan report for 192.168.100.1
Host is up (0.042s latency).
Nmap scan report for 192.168.100.12
Host is up (0.21s latency).
Nmap scan report for 192.168.100.13
Host is up (0.21s latency).
Nmap scan report for 192.168.100.15
Host is up (0.10s latency).
Nmap scan report for 192.168.100.16
Host is up (0.19s latency).
Nmap scan report for 192.168.100.25
Host is up (0.071s latency).
Nmap scan report for 192.168.100.28
Host is up (1.5s latency).
Nmap scan report for 192.168.100.7
Host is up.
Nmap done: 256 IP addresses (8 hosts up) scanned in 9.39 seconds
```

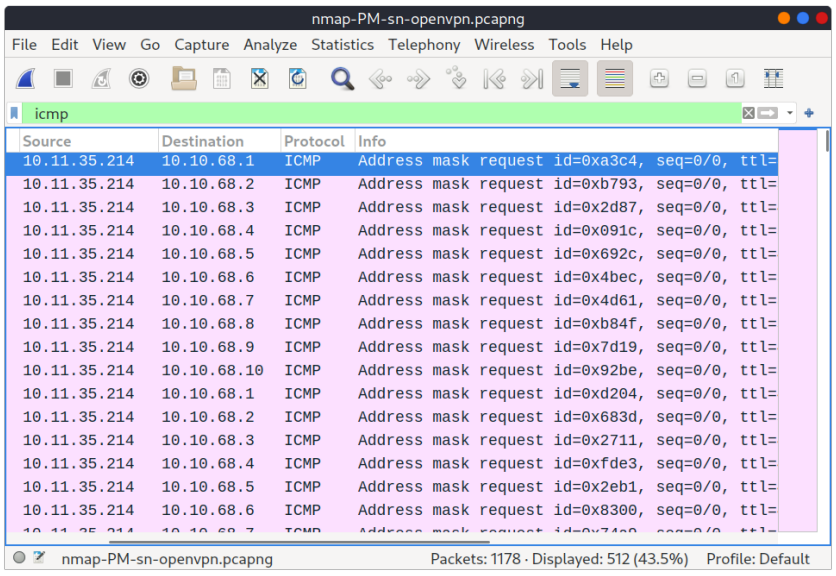


Figure 4: Wireshark capture of Address Mask Request packets.

Bonus: if you were to NOT specify one of the techniques above, just used ***nmap -sn*** with no other flags, Nmap's default behaviour would be to reverse-DNS online hosts. And to skip that, you can use ***-n*** flag.

2.3 TCP/UDP Host Discovery

2.3.I TCP SYN Scan (-PS)

Nmap will send TCP SYN packets and won't complete the TCP 3-way handshake even if the port is open, as shown in the figure below the terminal's output. By default, Nmap will send the SYN packet to port 80.

TCP SYN Scan (-PS)

```
$ 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
```

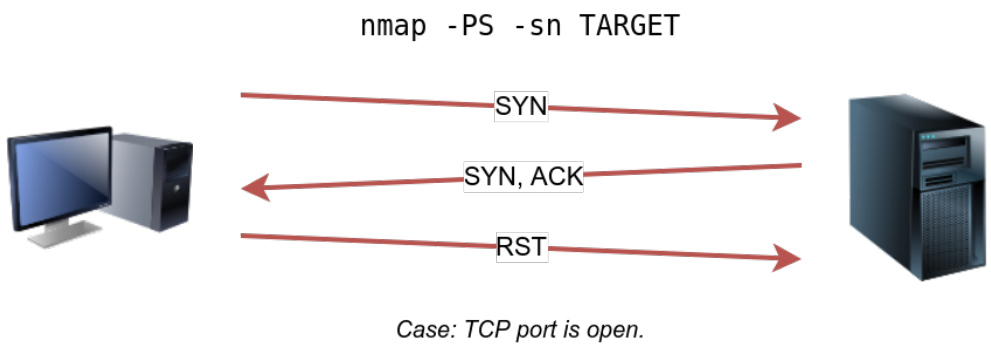


Figure 1: SYN Packet sent for Host Discovery, the user has root privileges.

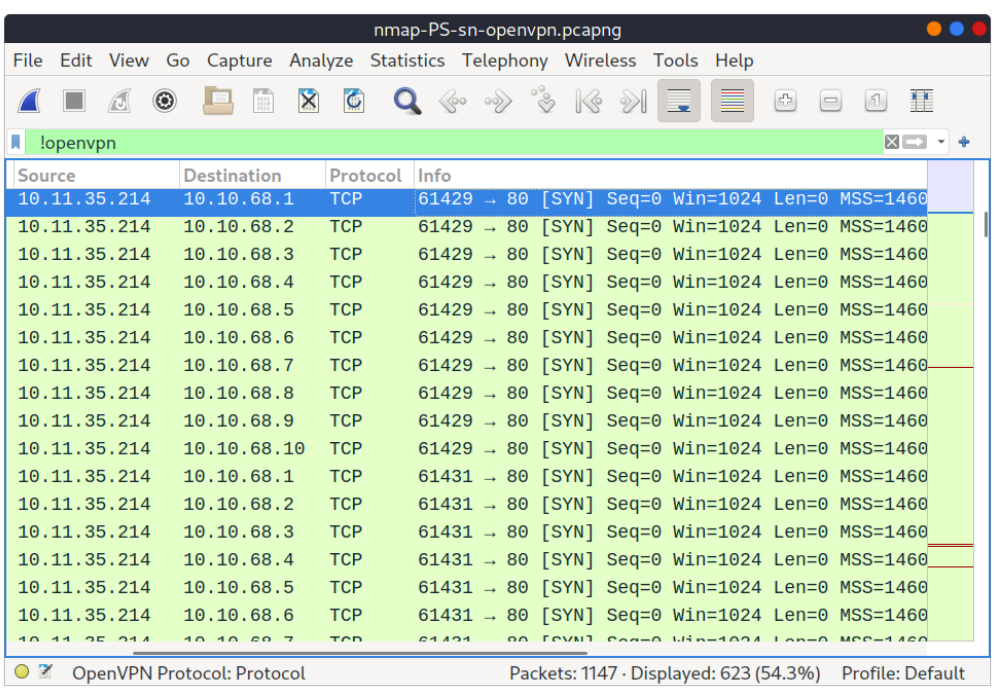


Figure 2: Do you notice how Nmap is sending the SYN packets to port 80? Twice? For each IP in the specified subnet.

2.3.II TCP ACK Scan (-PA)

As you have guessed, this sends a packet with an ACK flag set. Again, by default, Nmap will send the ACK packet to port 80.

TCP ACK Scan (-PA)

```
$ 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
```

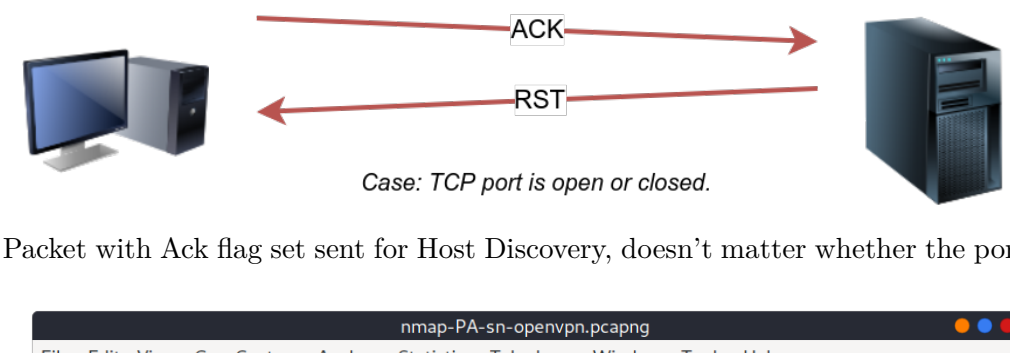
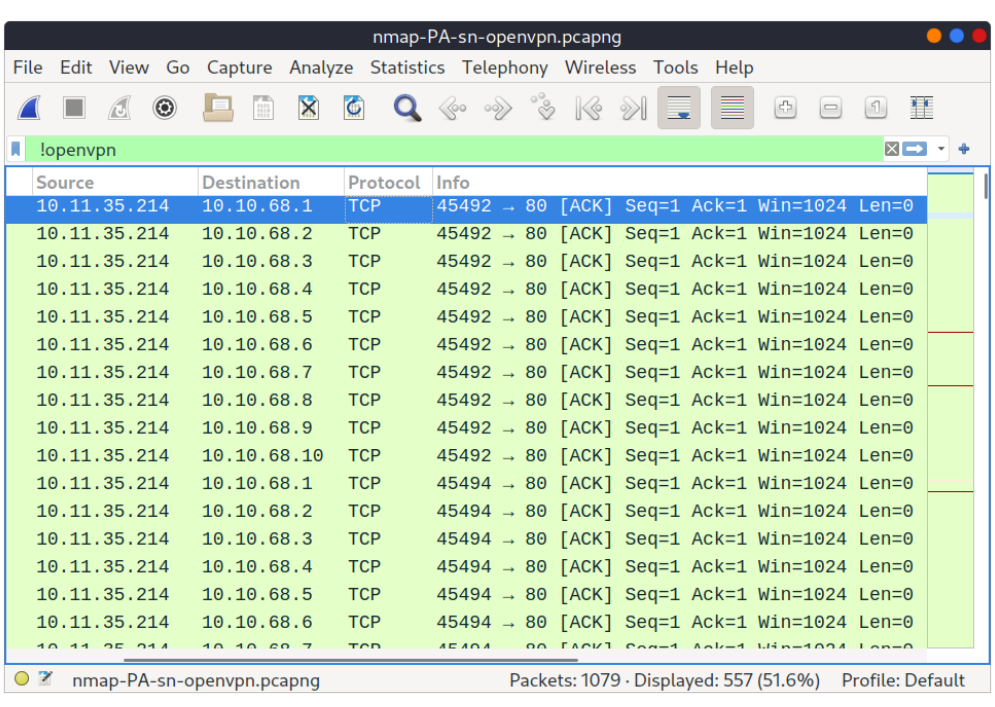


Figure 3: TCP Packet with Ack flag set sent for Host Discovery, doesn't matter whether the port is open or closed



2.3.III UDP Scan (-PU)

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, and that's exactly why Nmap will be sending to uncommon UDP ports, so that an ICMP destination unreachable is triggered.

UDP Scan (-PU)

```
$ 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
```

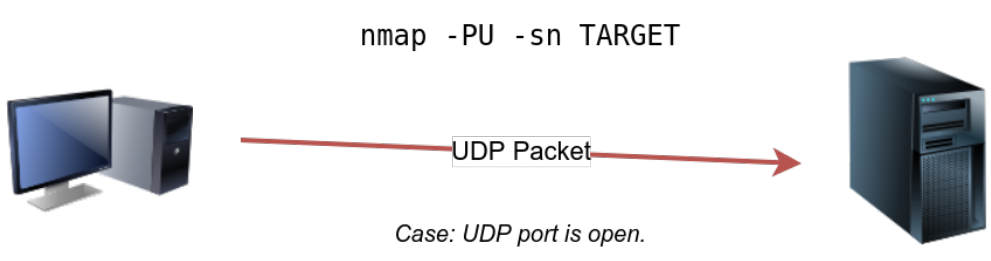


Figure 4: UDP Packet sent for Host Discovery, port is open and we don't get any reply.

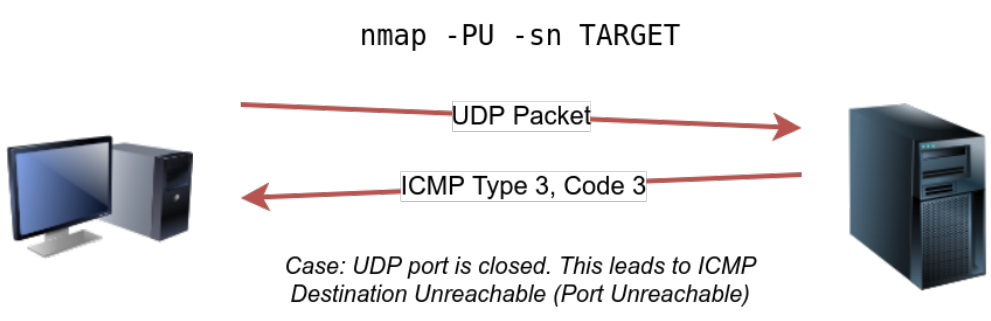


Figure 5: UDP Packet sent for Host Discovery, port is closed and we get a port unreachable packet.

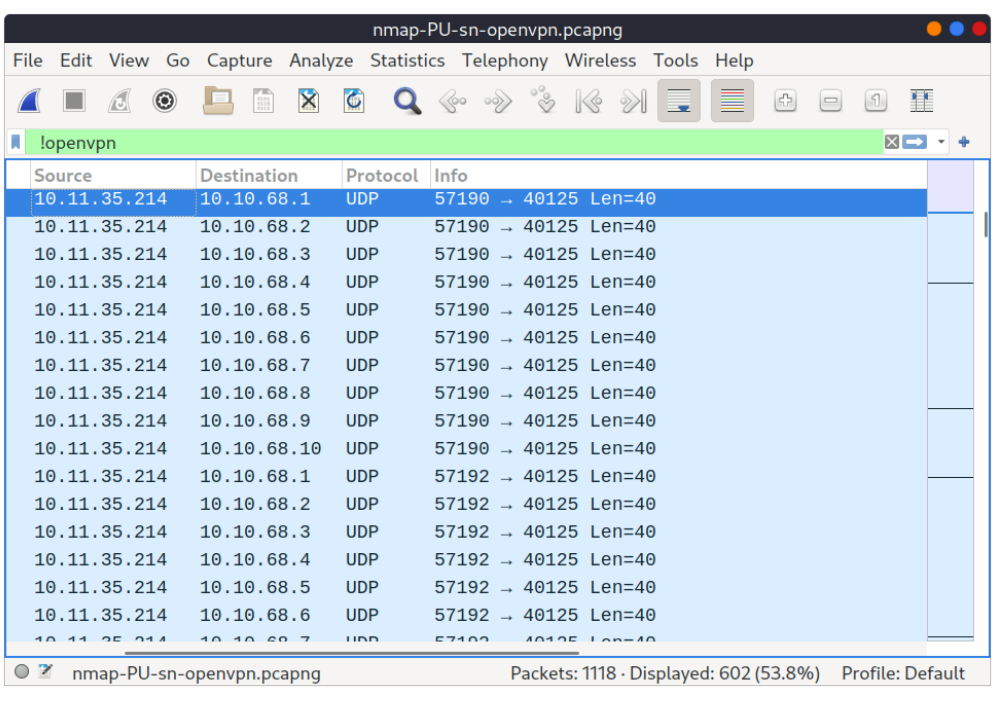


Figure 6: Do you notice how Nmap is sending the UDP packets to uncommon ports multiple times for each IP address in the specified subnet?

4 Basic Port Scanning

So far when we scanned ports we've seen only the **Open** state, but here's what Nmap also considers:

- **Open**: indicates that a service is listening on the specified port.
- **Closed**: indicates that no service is listening on the specified port, although the port is accessible. By *accessible*, we mean that it is reachable and is not blocked by a firewall or other security appliances/programs.
- **Filtered**: means that Nmap cannot determine if the port is open or closed because the port is not accessible. This state is usually due to a firewall preventing Nmap from reaching that port. Nmap's packets may be blocked from reaching the port; alternatively, the responses are blocked from reaching Nmap's host.
- **Unfiltered**: means that Nmap cannot determine if the port is open or closed, although the port is accessible. This state is encountered when using an ACK scan `-sA`.
- **Open|Filtered**: This means that Nmap cannot determine whether the port is open or filtered.
- **Closed|Filtered**: This means that Nmap cannot decide whether a port is closed or filtered.

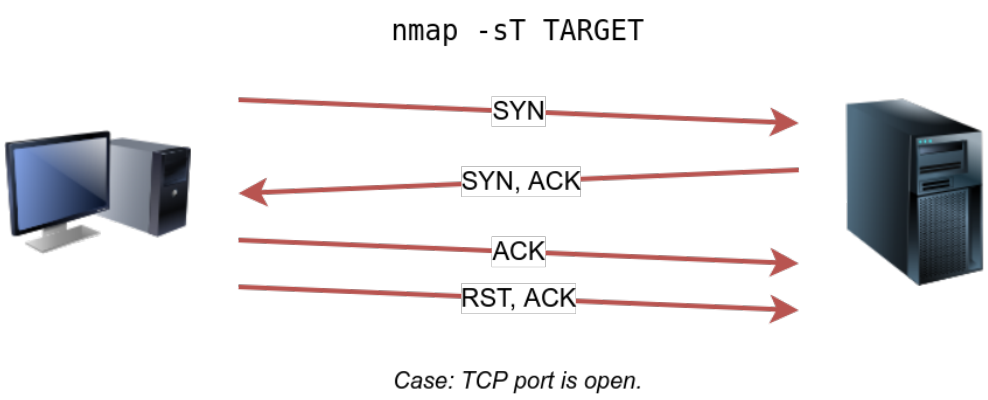


Figure 1: A closed TCP port responds to a SYN packet with RST/ACK to indicate that it is not open.

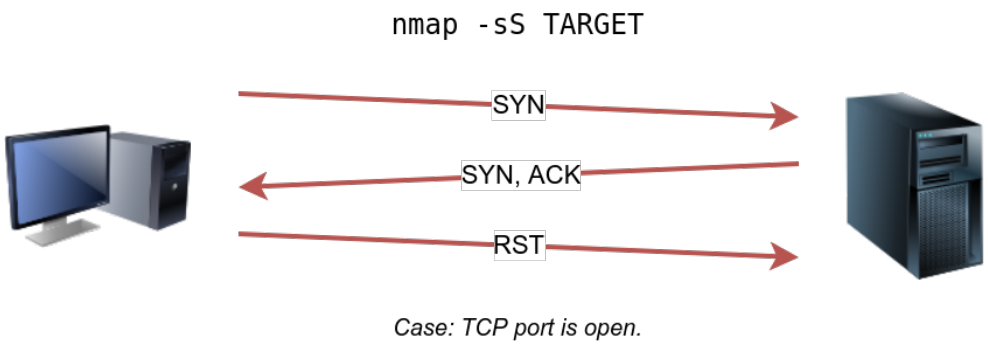


Figure 2: SYN scan does not need to complete the TCP 3-way handshake; instead, it tears down the connection once it receives a response from the server.

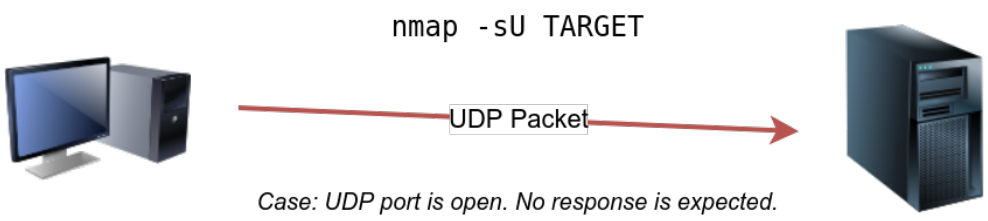


Figure 3: the UDP ports that don't generate any response are the ones that Nmap will state as open.

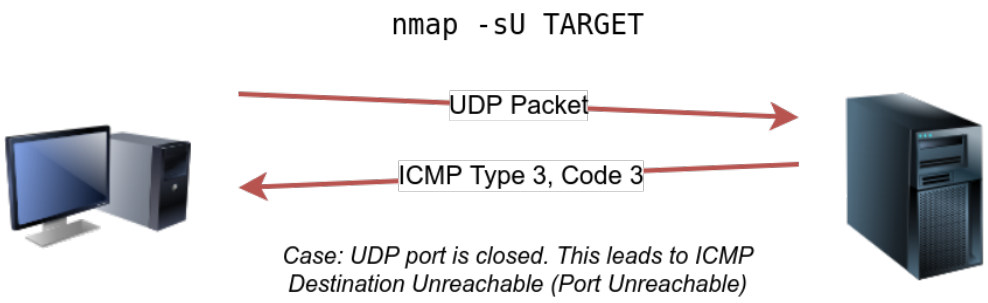
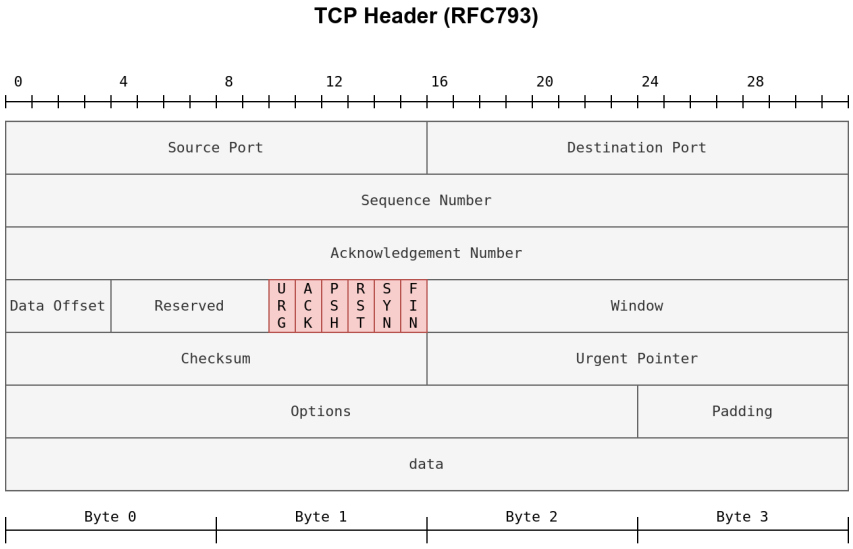


Figure 4: ICMP port unreachable message indicates the UDP port is closed.

4 Advanced Port Scanning

Now we'll need to revise the TCP header as these scans revolve around manipulating its flags.



Do you notice the flags in the fourth row? read em' vertically from top to bottom if you're a little confused.

4.1 Null Scan (-sN)

The null scan does not set any flag; all six flag bits are set to zero.

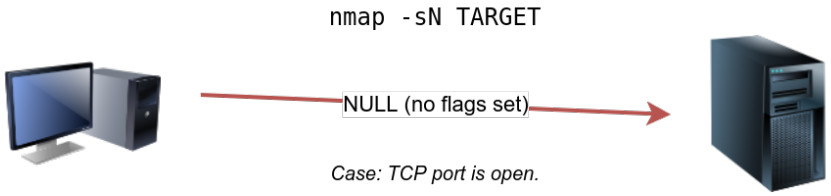


Figure 5: From Nmap's perspective, a lack of reply in a null scan indicates that either the port is open or a firewall is blocking the packet.

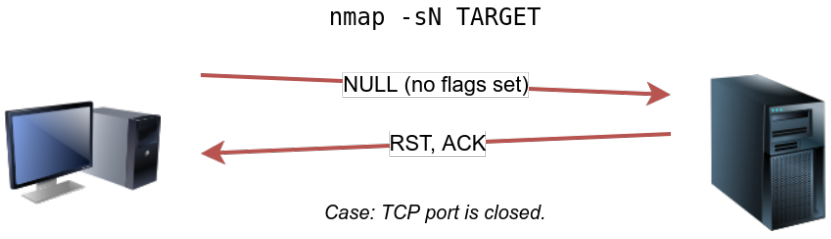


Figure 6: We expect the target server to respond with an RST packet if the port is closed.

4.2 FIN Scan (-sF)

The FIN scan sends a TCP packet with the FIN flag set.

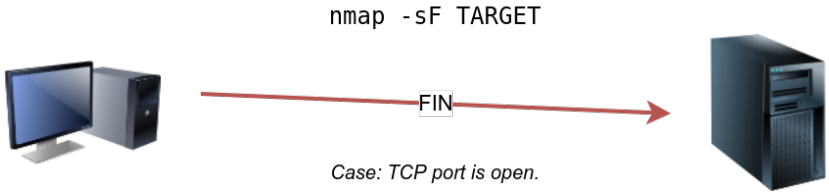


Figure 7: Similarly, no response will be sent if the TCP port is open. Again, Nmap cannot be sure if the port is open or if a firewall is blocking the traffic related to this TCP port.

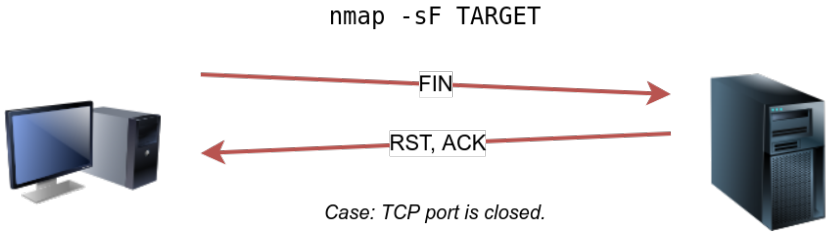
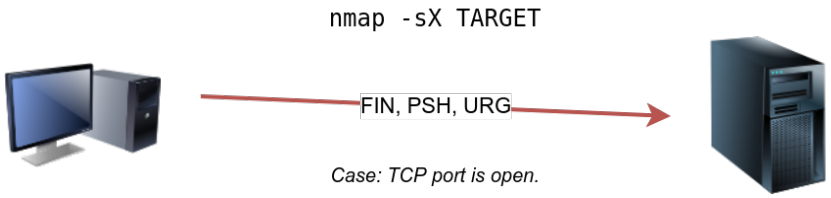
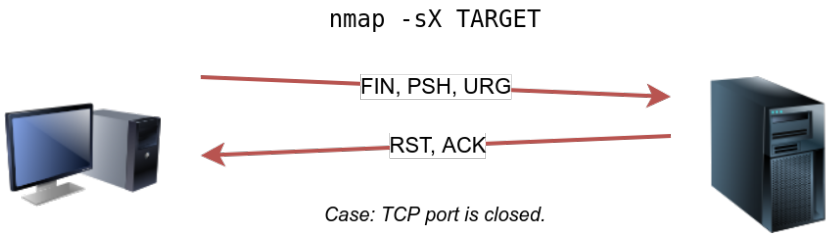


Figure 8: However, the target system should respond with an RST if the port is closed. Consequently, we will be able to know which ports are closed and use this knowledge to infer the ports that are open or filtered. It's worth noting some firewalls will 'silently' drop the traffic without sending an RST.

4.3 Xmas Scan (-sX)

The Xmas scan gets its name after Christmas tree lights. An Xmas scan sets the FIN, PSH, and URG flags simultaneously. 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.





4.4 Ack Scan (-sA)

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.

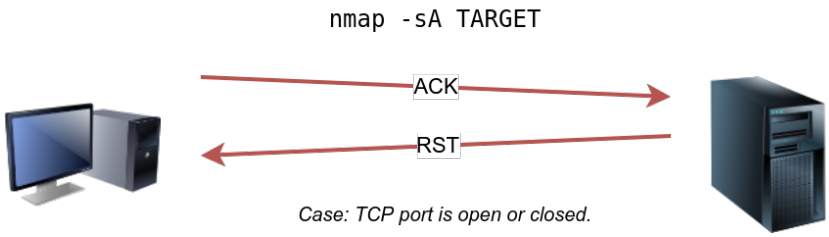


Figure 9: This kind of scan would be helpful if there is a firewall in front of the target. Consequently, based on which ACK packets resulted in responses, you will learn which ports were not blocked by the firewall. In other words, this type of scan is more suitable to discover firewall rule sets and configuration.

4.5 Window Scan (-sW)

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.

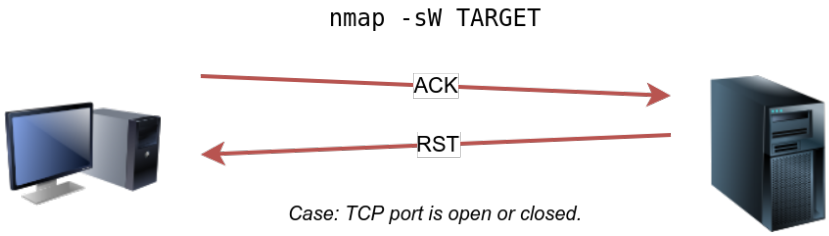
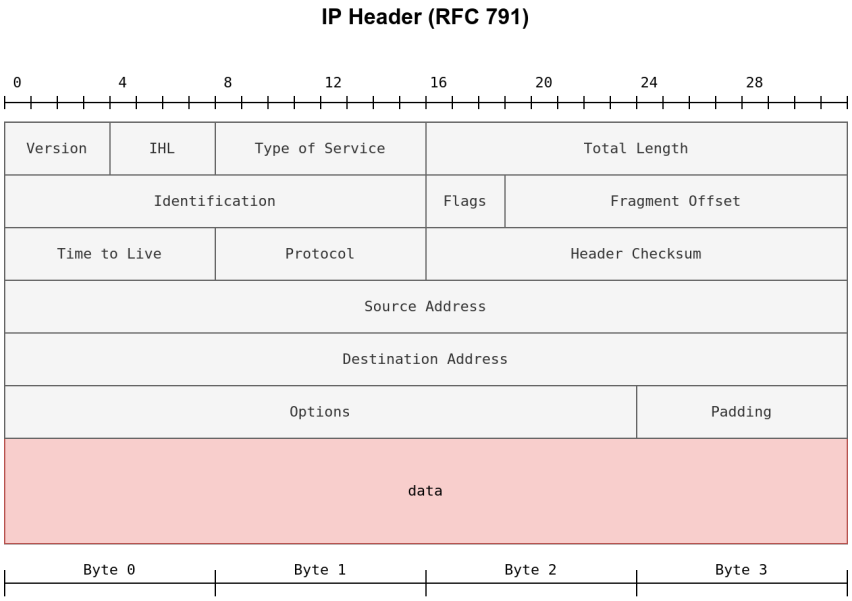


Figure 10: This is also useful only when there's a firewall on the target system.

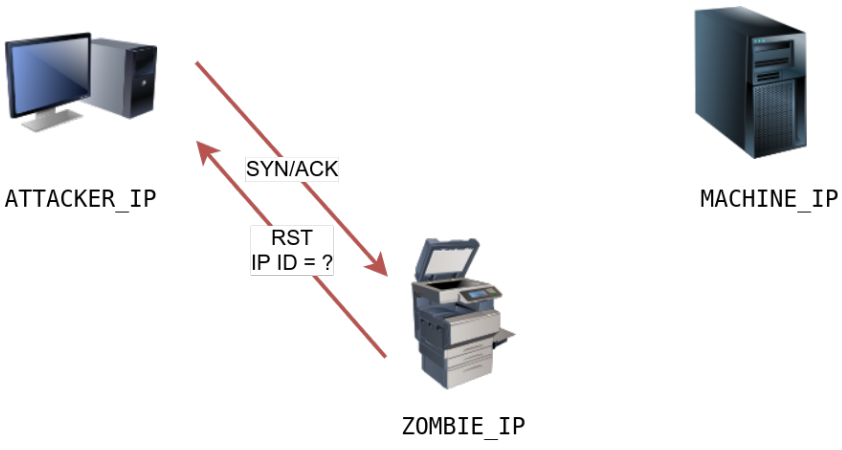
5 Fragmented Packets (-f)

We use this technique in an attempt to evade Firewalls and IDSs. 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.



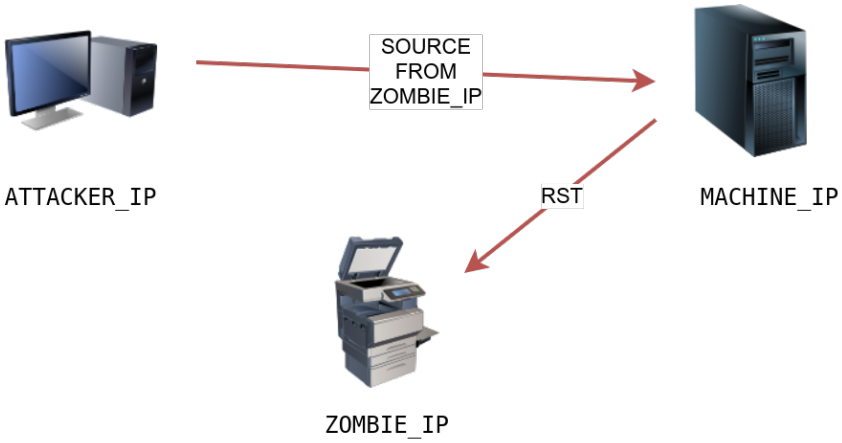
6 Zombie Scan (-sI)

One can't always benefit from spoofing his IP address via `nmap -S SPOOFED_IP MACHINE_IP`, and that's simply because you won't always be able to monitor the target networks traffic. And that's where zombie scans come in handy. In the figure below, we have the attacker system probing an idle machine, a multi-function printer. By sending a SYN/ACK, it responds with an RST packet containing its newly incremented IP ID.



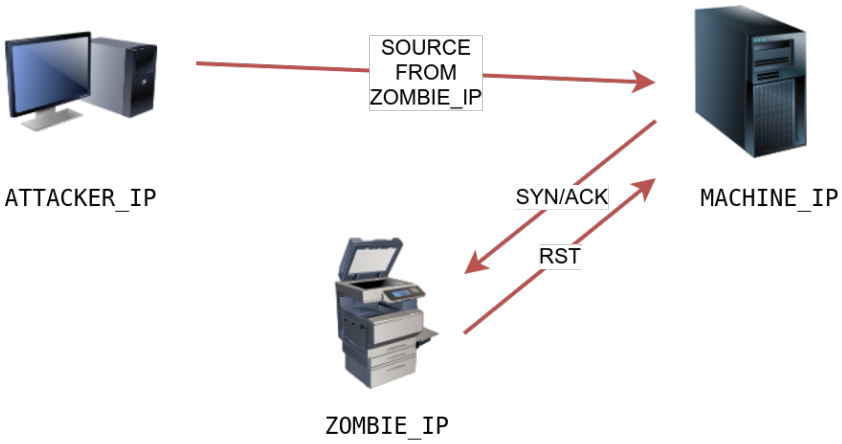
Attacker system communicates with an idle system to find its current IP ID

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 closed

In the second scenario, as shown below, the TCP port is open, so the target machine responds with a SYN/ACK to the idle host (zombie). The idle host responds to this unexpected packet with an RST packet, thus incrementing its IP ID.



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.