Firewall and IDS/IPS Evasion

Determine Firewalls and Their Rules

We already know that when a port is shown as **filtered**, it can have several reasons. In most cases, firewalls have certain rules set to handle specific connections. The packets can either be dropped, or rejected. **The dropped packets** are ignored, and **no response is returned from the host.**

This is different for **rejected packets** that **are returned with an RST flag**. These packets **contain different types of ICMP error** codes or contain nothing at all.

Such errors can be:

- Net Unreachable
- Net Prohibited
- Host Unreachable
- Host Prohibited
- Port Unreachable
- Proto Unreachable

Nmap's TCP ACK scan (-sA) method **is much harder to filter for firewalls and IDS/IPS systems** than regular SYN (-sS) or Connect scans (sT) **because they only send a TCP packet with only the ACK flag**. When a port is closed or open, **the host must respond with an RST flag**.

Unlike outgoing connections, all connection attempts (with the SYN flag) from external networks are usually blocked by firewalls. However, **the packets with the ACK flag are often passed by the firewall** because the firewall cannot determine whether the connection was first established from the external network or the internal network.

SYN-Scan

Firewall and IDS/IPS Evasion

```
Djerbien@htb[/htb]$ sudo nmap 10.129.2.28 -p 21,22,25 -sS -Pn -n --disable-arp-ping --packet-trace

Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-21 14:56 CEST
SENT (0.0278s) TCP 10.10.14.2:57347 > 10.129.2.28:22 S ttl=53 id=22412 iplen=44  seq=4092255222 win=1024 <m
SENT (0.0278s) TCP 10.10.14.2:57347 > 10.129.2.28:25 S ttl=50 id=62291 iplen=44  seq=4092255222 win=1024 <m
SENT (0.0278s) TCP 10.10.14.2:57347 > 10.129.2.28:21 S ttl=58 id=38696 iplen=44  seq=4092255222 win=1024 <m
RCVD (0.0329s) ICMP [10.129.2.28 > 10.10.14.2 Port 21 unreachable (type=3/code=3) ] IP [ttl=64 id=40884 ipl
RCVD (0.0341s) TCP 10.129.2.28:22 > 10.10.14.2:57347 SA ttl=64 id=0 iplen=44  seq=1153454414 win=64240 <mss
RCVD (1.0386s) TCP 10.129.2.28:22 > 10.10.14.2:57347 SA ttl=64 id=0 iplen=44  seq=1153454414 win=64240 <mss
SENT (1.1366s) TCP 10.10.14.2:57348 > 10.129.2.28:25 S ttl=44 id=6796 iplen=44  seq=4092320759 win=1024 <m
Nmap scan report for 10.129.2.28
Host is up (0.0053s latency).

PORT   STATE    SERVICE
21/tcp filtered ftp
22/tcp open     ssh
25/tcp filtered smtp
MAC Address: DE:AD:00:00:BE:EF (Intel Corporate)

Nmap done: 1 IP address (1 host up) scanned in 0.07 seconds
```

ACK-Scan

Firewall and IDS/IPS Evasion

```
Djerbien@htb[/htb]$ sudo nmap 10.129.2.28 -p 21,22,25 -sA -Pn -n --disable-arp-ping --packet-trace

Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-21 14:57 CEST
SENT (0.0422s) TCP 10.10.14.2:49343 > 10.129.2.28:21 A ttl=49 id=12381 iplen=40  seq=0 win=1024
SENT (0.0423s) TCP 10.10.14.2:49343 > 10.129.2.28:22 A ttl=41 id=5146 iplen=40  seq=0 win=1024
SENT (0.0423s) TCP 10.10.14.2:49343 > 10.129.2.28:25 A ttl=49 id=5800 iplen=40  seq=0 win=1024
RCVD (0.1252s) ICMP [10.129.2.28 > 10.10.14.2 Port 21 unreachable (type=3/code=3) ] IP [ttl=64 id=55628 ipl
RCVD (0.1268s) TCP 10.129.2.28:22 > 10.10.14.2:49343 R ttl=64 id=0 iplen=40  seq=1660784500 win=0
SENT (1.3837s) TCP 10.10.14.2:49344 > 10.129.2.28:25 A ttl=59 id=21915 iplen=40  seq=0 win=1024
Nmap scan report for 10.129.2.28
Host is up (0.083s latency).

PORT   STATE      SERVICE
21/tcp filtered   ftp
22/tcp unfiltered ssh
25/tcp filtered   smtp
MAC Address: DE:AD:00:00:BE:EF (Intel Corporate)

Nmap done: 1 IP address (1 host up) scanned in 0.15 seconds
```

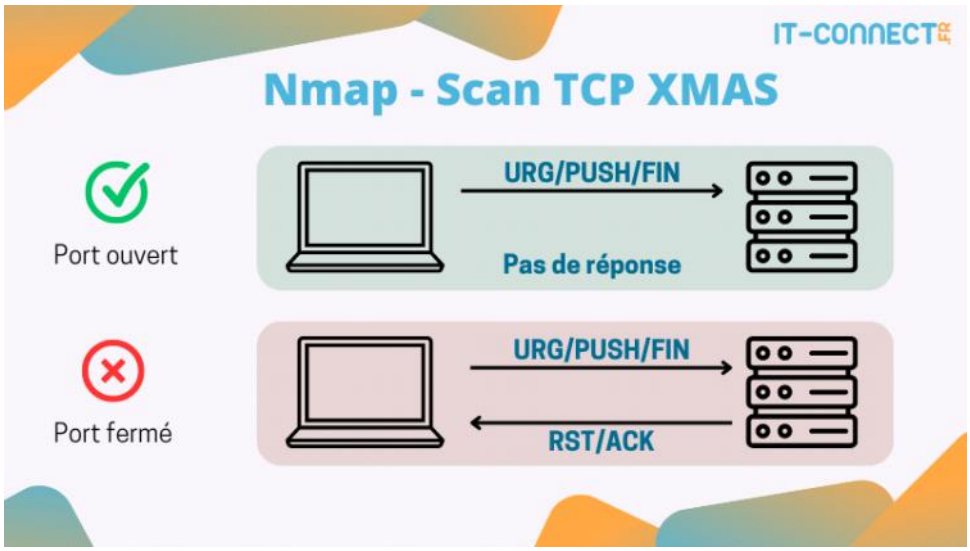
II. Le TCP XMAS scan

Le TCP XMAS Scan est un peu particulier, car il ne simule pas du tout un comportement normal d'utilisateur ou de machine au sein d'un réseau. En effet, le TCP XMAS Scan va envoyer des paquets TCP avec les flags "URG" (Urgent), "PSH" (Push), et "FIN" (Finish) à 1 dans le but de déjouer certains pare-feu ou mécanismes de filtrage.

```
Flags: 0x029 (FIN, PSH, URG)
   000. .... .... = Reserved: Not set
   ...0 .... .... = Accurate ECN: Not set
   .... 0... .... = Congestion Window Reduced: Not set
   .... .0.. .... = ECN-Echo: Not set
   .... ..1. .... = Urgent: Set
   .... ...0 .... = Acknowledgment: Not set
   .... .... 1... = Push: Set
   .... .... .0.. = Reset: Not set
   .... .... ..0. = Syn: Not set
   .... .... ...1 = Fin: Set
```

Sans détailler le rôle de ces flags ici, **il faut savoir que lors d'un envoi de paquet avec ces trois flags activés**, **un service actif** derrière le port visé **ne renverra aucun paquet**.

En revanche, **si le port est fermé**, **nous recevrons un paquet TCP RST/ACK**. On saura alors différencier le comportement d'un port ouvert et d'un port fermé pour lister les ports sur une machine :



Exemple si on envoie à un site web :

No.	Source	Destination	Protocol	Info
1	10.10.14.84	10.10.10.203	TCP	64540 → 80 [FIN, PSH, URG]
2	10.10.14.84	10.10.10.203	TCP	64542 → 80 [FIN, PSH, URG]

On obtient aucun retour de server 10.10.10.203 sur le port 80 ce qui indique qu'il est ouvert

Par contre :

No.	Source	Destination	Protocol	Info
1	10.10.14.84	10.10.10.100	TCP	59128 → 80 [FIN, PSH, URG]
2	10.10.10.100	10.10.14.84	TCP	80 → 59128 [RST, ACK]

Nous voyons donc que la réponse à notre paquet TCP est un TCP RST/ACK, le port est donc fermé. En utilisant Nmap, c'est l'option "-sX" (scan XMAS qui permet d'effectuer un TCP XMAS Scan :

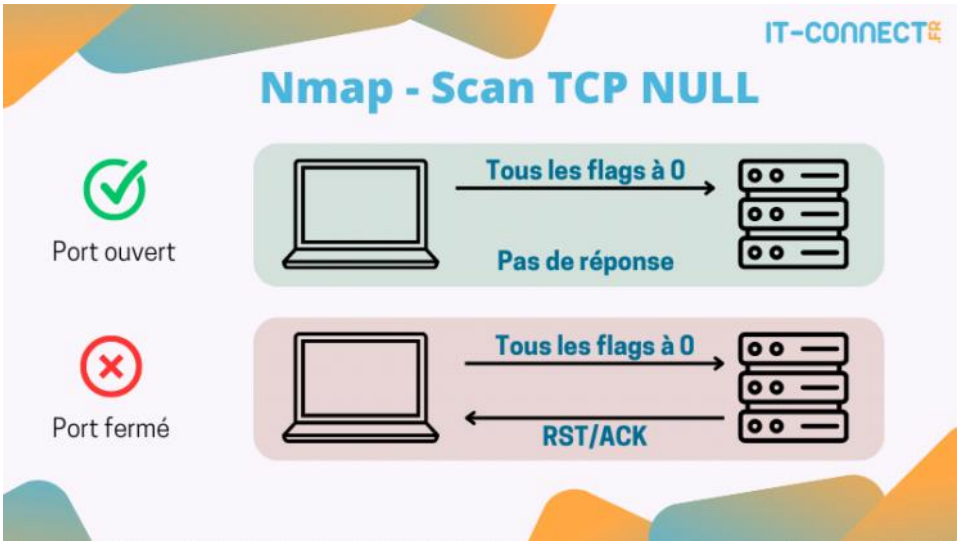
```
nmap -sX 192.168.1.15
```

Il est important de signaler que le TCP XMAS scan n'est pas capable de détecter les pare-feu qui pourraient se trouver entre la cible et la machine de scan à l'inverse de certains autres types de scans comme le TCP SYN ou Connect. En effet, un pare-feu qui serait actif entre les deux hôtes ferait en sorte qu'aucun retour TCP ne soit fait si le port visé est filtré (c'est-à-dire protégé par le pare-feu). On est alors dans l'impossibilité, lors d'une non-réponse, de savoir s'il s'agit d'un port protégé par le pare-feu ou alors d'un port ouvert et actif. Il faut également savoir que, comme le TCP FIN scan décrit dans le chapitre, certaines applications ou OS comme les OS Windows peuvent fausser les résultats de ce type de scan.

III. Le TCP Null scan

À l'inverse du TCP XMAS scan, le TCP Null scan va envoyer des paquets TCP scan avec tous les flags à 0. Il s'agit là aussi d'un comportement que l'on ne trouvera jamais dans un échange normal entre des machines, car l'envoi d'un paquet TCP sans flag n'est pas spécifié dans le RFC décrivant le protocole TCP. C'est pourquoi il peut être détecté plus facilement.

L'utilisation de ce scan peut, comme le TCP XMAS scan, perturber certains firewalls ou modules de filtrage et alors laisser passer les paquets :



Voici ce que l'on peut observer sur le réseau **lorsque l'on effectue un TCP Null scan sur un port ouvert :**

No.	Source	Destination	Protocol	Info
1	10.10.14.84	10.10.10.203	TCP	33742 → 80 [<None>]
2	10.10.14.84	10.10.10.203	TCP	33744 → 80 [<None>]

On voit que la machine de scan envoie un paquet sans flag (qui se traduit par l'inscription [<None>] dans Wireshark) sans aucune réponse du serveur. À l'inverse, voici ce qu'il se passe lorsque le port visé est fermé :

No.	Source	Destination	Protocol	Info
1	10.10.14.84	10.10.10.100	TCP	52644 → 80 [<None>]
2	10.10.10.100	10.10.14.84	TCP	80 → 52644 [RST, ACK]

En utilisant Nmap, c'est l'option "-sN" (scan Null) qui permet d'effectuer un TCP NULL Scan :

```
nmap -sN 192.168.1.15
```

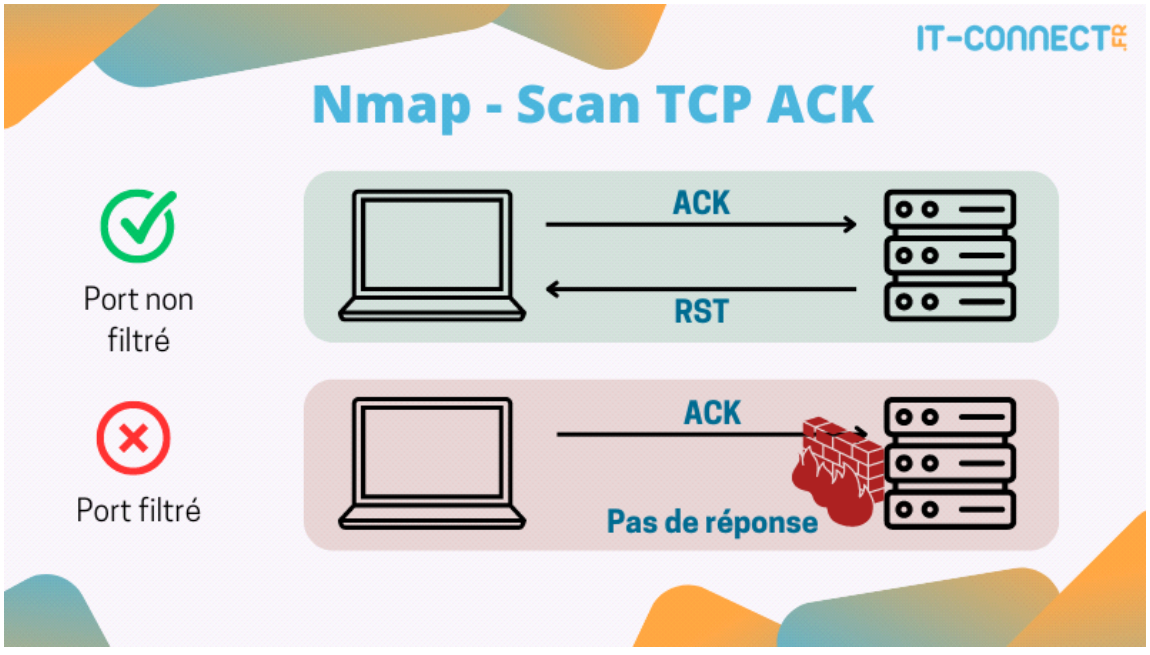
- Étant donné que la réponse quand un port est ouvert et quand un firewall est actif (c'est-à-dire aucun retour du serveur dans les deux cas), le TCP Null scan est incapable de détecter la présence d'un pare-feu. De plus, le pare-feu peut même fausser le résultat en faisant croire qu'un port est ouvert, car il ne répond pas aux paquets TCP sans flags alors que celui-ci est filtré et donc protégé. C'est une information à connaître lorsque l'on emploie des scans qui sont incapables de différencier un port ouvert d'un port filtré (protégé par un pare-feu) comme les scans TCP Null, XMAS ou FIN,afin de rester cohérent dans l'interprétation des résultats obtenus.

IV. Le TCP ACK scan

- **Le TCP ACK scan est utilisé pour détecter la présence d'un pare-feu sur l'hôte cible ou entre la cible et la source de scan.**

En effet, **contrairement aux autres scans**, le TCP ACK scan ne va pas avoir pour objectif de voir quel port est ouvert sur l'hôte, mais **plutôt de savoir si un système de filtrage est actif** en répondant pour chaque port par "**filtered**" ou "**unfiltered**".

Certains scans TCP comme les TCP SYN ou TCP Connect savent faire les deux en même temps alors que d'autres comme TCP FIN ou TCP XMAS ne permettent pas du tout de déterminer la présence d'un filtrage, c'est pourquoi le TCP ACK scan peut avoir un intérêt :



On utilisera l'option "**-sA**" de l'outil nmap pour effectuer ce type de scan, voici le résultat que nous pourrons avoir lors de

l'**exécution de ce TCP ACK scan dans le cas où le port est filtré**, c'est-à-dire bloqué et protégé par un pare-feu :

```
┌──(mickael㉿kali-it-connect)-[~/Downloads]
└─$ sudo nmap -p 80,81  10.10.10.100 10.10.10.203 -sA -Pn
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-06 21:18
Nmap scan report for 10.10.10.100
Host is up (0.018s latency).

PORT   STATE      SERVICE
80/tcp unfiltered http
81/tcp unfiltered hosts2-ns

Nmap scan report for 10.10.10.203
Host is up.

PORT   STATE    SERVICE
80/tcp filtered http
81/tcp filtered hosts2-ns

Nmap done: 2 IP addresses (2 hosts up) scanned in 1.30 seconds
```

Nous voyons ici un exemple de résultat pour un hôte disposant d'un pare-feu et un autre n'en disposant pas. Nmap retourne un "filtered" sur les ports TCP/80 et TCP/81 de l'hôte "10.10.10.203". Sur une analyse réseau via Wireshark, nous pourrions voir le comportement suivant :

No.	Source	Destination	Protocol	Info
1	10.10.14.84	10.10.10.203	TCP	52553 → 80 [ACK]
2	10.10.14.84	10.10.10.203	TCP	52553 → 81 [ACK]
3	10.10.14.84	10.10.10.203	TCP	52555 → 81 [ACK]
4	10.10.14.84	10.10.10.203	TCP	52555 → 80 [ACK]

On voit donc que la machine cible nous renvoie un paquet TCP RST dans le cas où aucun pare-feu n'est présent.

En utilisant Nmap, c'est l'option "-sA" (scan ACK qui permet d'effectuer un TCP ACK Scan :

```
nmap -sA 192.168.1.15
```

```
sudo nmap 10.129.2.28 -p 21,22,25 -sA -Pn -n --disable-arp-ping --packet-trace
```

Detect IDS/IPS

Decoys

There are cases in which administrators block specific subnets from different regions in principle. This prevents any access to the target network. Another example is when IPS should block us. For this reason, the Decoy scanning method **(-D)** is the right choice. With this method, **Nmap generates various random IP addresses inserted into the IP header to disguise the origin of the packet sent**. With this method, we can generate random **(RND)** a specific number (for example**: 5**) of IP addresses separated by a colon (:). Our real IP address is then randomly placed between the generated IP addresses. In the next example, our real IP address is therefore placed in the second position. Another critical point is that the decoys must be alive. Otherwise, the service on the target may be unreachable due to SYN-flooding security mechanisms.

Scan by Using Decoys

Firewall and IDS/IPS Evasion

```
Djerbien@htb[/htb]$ sudo nmap 10.129.2.28 -p 80 -sS -Pn -n --disable-arp-ping --packet-trace -D RND:5

Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-21 16:14 CEST
SENT (0.0378s) TCP 102.52.161.59:59289 > 10.129.2.28:80 S ttl=42 id=29822 iplen=44  seq=3687542010 win=1024
SENT (0.0378s) TCP 10.10.14.2:59289 > 10.129.2.28:80 S ttl=59 id=29822 iplen=44  seq=3687542010 win=1024 <m
SENT (0.0379s) TCP 210.120.38.29:59289 > 10.129.2.28:80 S ttl=37 id=29822 iplen=44  seq=3687542010 win=1024
SENT (0.0379s) TCP 191.6.64.171:59289 > 10.129.2.28:80 S ttl=38 id=29822 iplen=44  seq=3687542010 win=1024 
SENT (0.0379s) TCP 184.178.194.209:59289 > 10.129.2.28:80 S ttl=39 id=29822 iplen=44  seq=3687542010 win=10
SENT (0.0379s) TCP 43.21.121.33:59289 > 10.129.2.28:80 S ttl=55 id=29822 iplen=44  seq=3687542010 win=1024 
RCVD (0.1370s) TCP 10.129.2.28:80 > 10.10.14.2:59289 SA ttl=64 id=0 iplen=44  seq=4056111701 win=64240 <mss
Nmap scan report for 10.129.2.28
Host is up (0.099s latency).

PORT   STATE SERVICE
80/tcp open  http
MAC Address: DE:AD:00:00:BE:EF (Intel Corporate)

Nmap done: 1 IP address (1 host up) scanned in 0.15 seconds
```

The spoofed packets are often filtered out by ISPs and routers, even though they come from the same network range. Therefore, we can also specify our VPS servers' IP addresses and use them in combination with "IP ID" manipulation in the IP headers to scan the target.

Another scenario would be that only individual subnets would not have access to the server's specific services. So we can also manually specify the source IP address (-S) to test if we get better results with this one. Decoys can be used for SYN, ACK, ICMP scans, and OS detection scans. So let us look at such an example and determine which operating system it is most likely to be.

.

Testing Firewall Rule

Firewall and IDS/IPS Evasion

```
Djerbien@htb[/htb]$ sudo nmap 10.129.2.28 -n -Pn -p445 -O

Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-22 01:23 CEST
Nmap scan report for 10.129.2.28
Host is up (0.032s latency).

PORT    STATE    SERVICE
445/tcp filtered microsoft-ds
MAC Address: DE:AD:00:00:BE:EF (Intel Corporate)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 3.14 seconds
```

Scan by Using Different Source IP

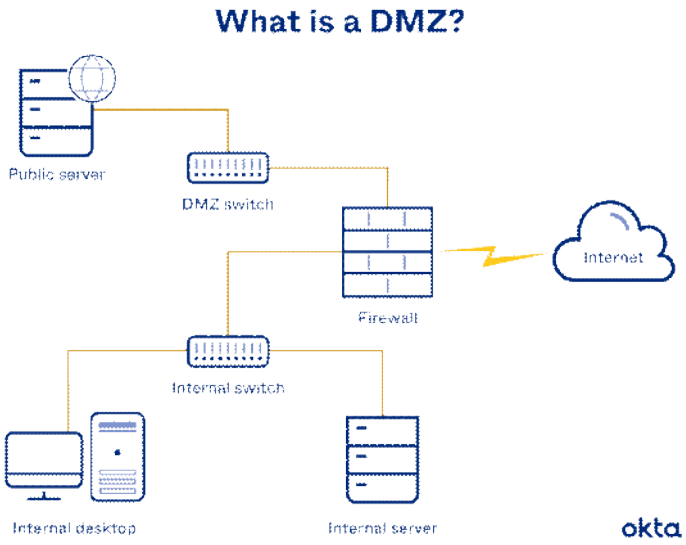
Firewall and IDS/IPS Evasion

```
Djerbien@htb[/htb]$ sudo nmap 10.129.2.28 -n -Pn -p 445 -O -S 10.129.2.200 -e tun0

Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-22 01:16 CEST
Nmap scan report for 10.129.2.28
Host is up (0.010s latency).

PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: DE:AD:00:00:BE:EF (Intel Corporate)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 2.6.32 (96%), Linux 3.2 - 4.9 (96%), Linux 2.6.32 - 3.10 (96%), Linux 3.4 - 3.
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 4.11 seconds
```

DNS Proxying

By default, `Nmap` performs a reverse DNS resolution unless otherwise specified to find more important information about our target. These DNS queries are also passed in most cases because the given web server is supposed to be found and visited. The DNS queries are made over the `UDP port 53`. The `TCP port 53` was previously only used for the so-called "`Zone transfers`" between the DNS servers or data transfer larger than 512 bytes. More and more, this is changing due to IPv6 and DNSSEC expansions. These changes cause many DNS requests to be made via TCP port 53.

However, `Nmap` still gives us a way to specify DNS servers ourselves (`--dns-server <ns>,<ns>`). This method could be fundamental to us if we are in a demilitarized zone (`DMZ`). The company's DNS servers are usually more trusted than those from the Internet. So, for example, we could use them to interact with the hosts of the internal network. As another example, we can use `TCP port 53` as a source port (`--source-port`) for our scans. If the administrator uses the firewall to control this port and does not filter IDS/IPS properly, our TCP packets will be trusted and passed through.



D'un point de vue informatique, un réseau DMZ est un sous-réseau qui isole les services « publics » des services « privés ». Lorsqu'il est correctement implémenté, un réseau DMZ doit réduire le risque d'une brèche de données catastrophique. Les serveurs accessibles au public sont situés dans la zone démilitarisée, mais ils communiquent avec des bases de données

protégées par des pare-feux.

SYN-Scan of a Filtered Port

Firewall and IDS/IPS Evasion

```
Djerbien@htb[/htb]$ sudo nmap 10.129.2.28 -p50000 -sS -Pn -n --disable-arp-ping --packet-trace

Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-21 22:50 CEST
SENT (0.0417s) TCP 10.10.14.2:33436 > 10.129.2.28:50000 S ttl=41 id=21939 iplen=44  seq=736533153 win=1024 <mss 1460>
SENT (1.0481s) TCP 10.10.14.2:33437 > 10.129.2.28:50000 S ttl=46 id=6446 iplen=44  seq=736598688 win=1024 <mss 1460>
Nmap scan report for 10.129.2.28
Host is up.

PORT      STATE    SERVICE
50000/tcp filtered ibm-db2

Nmap done: 1 IP address (1 host up) scanned in 2.06 seconds
```

SYN-Scan From DNS Port

Firewall and IDS/IPS Evasion

```
Djerbien@htb[/htb]$ sudo nmap 10.129.2.28 -p50000 -sS -Pn -n --disable-arp-ping --packet-trace --source-port 53

SENT (0.0482s) TCP 10.10.14.2:53 > 10.129.2.28:50000 S ttl=58 id=27470 iplen=44  seq=4003923435 win=1024 <mss 1460>
RCVD (0.0608s) TCP 10.129.2.28:50000 > 10.10.14.2:53 SA ttl=64 id=0 iplen=44  seq=540635485 win=64240 <mss 1460>
Nmap scan report for 10.129.2.28
Host is up (0.013s latency).

PORT      STATE SERVICE
50000/tcp open  ibm-db2
MAC Address: DE:AD:00:00:BE:EF (Intel Corporate)

Nmap done: 1 IP address (1 host up) scanned in 0.08 seconds
```

Scanning Options	Description
`10.129.2.28`	Scans the specified target.
`-p 50000`	Scans only the specified ports.
`-sS`	Performs SYN scan on specified ports.
`-Pn`	Disables ICMP Echo requests.
`-n`	Disables DNS resolution.
`--disable-arp-ping`	Disables ARP ping.
`--packet-trace`	Shows all packets sent and received.
`--source-port 53`	Performs the scans from specified source port.

Now that we have found out that the firewall accepts `TCP port 53`, it is very likely that IDS/IPS filters might also be configured much weaker than others. We can test this by trying to connect to this port by using `Netcat`.

Connect To The Filtered Port

Firewall and IDS/IPS Evasion

```
Djerbien@htb[/htb]$ ncat -nv --source-port 53 10.129.2.28 50000

Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Connected to 10.129.2.28:50000.
220 ProFTPd
```

.