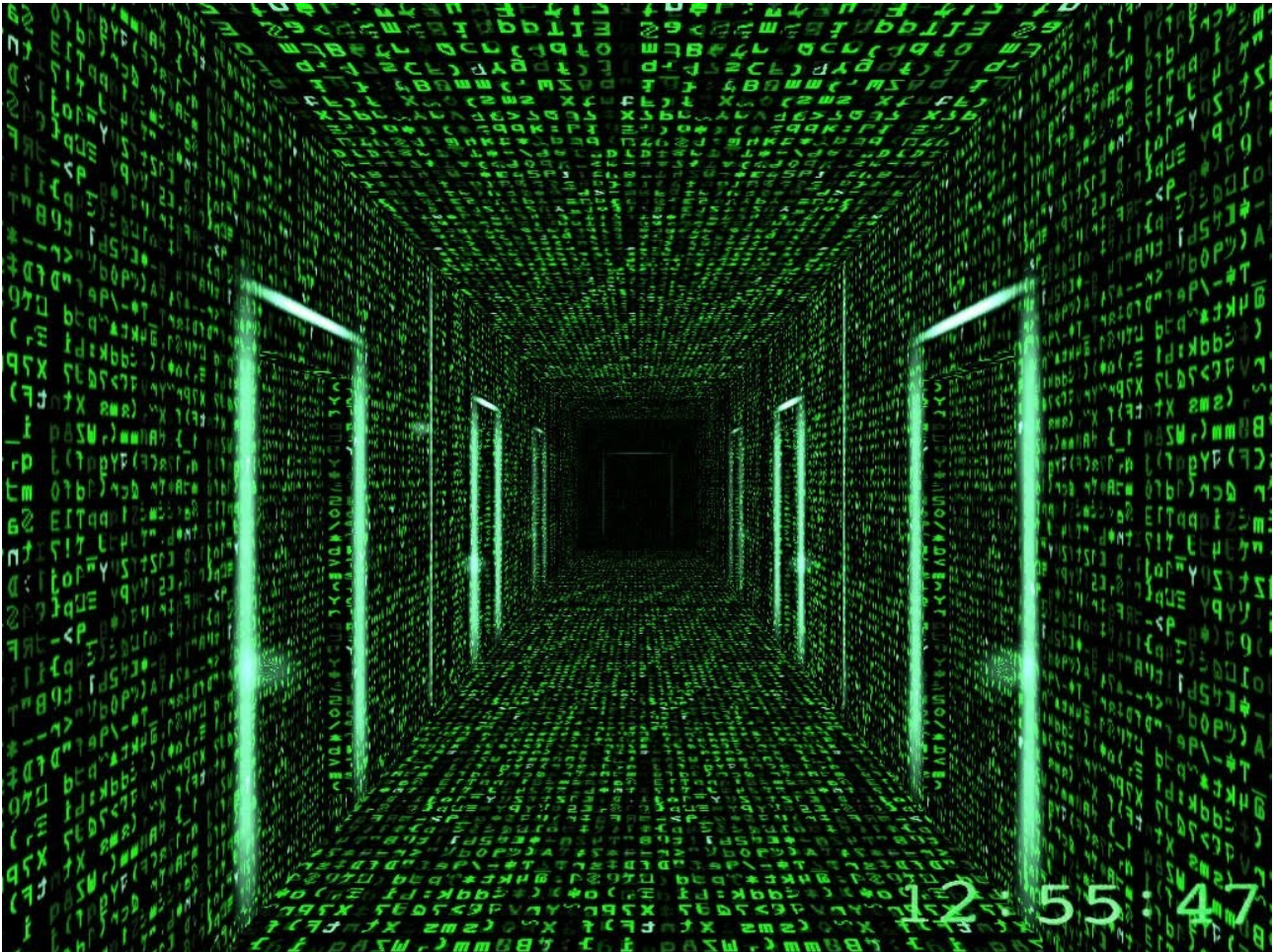


2.2 PORT SCANNING A HOST DESCONOCIDO



1. UN POCO DE TEORÍA

Antes de aprender como analizar los diferentes puertos de un equipo, debemos saber como se hace y con que finalidad.

Un **port scanning** es una técnica de information gathery que consiste en el envío de distintos tipos de tramas a los **puertos lógicos** de un equipo para averiguar cual es accesible, y posteriormente analizarlo, buscar vulnerabilidades y atacarlo.

Esta técnica a veces es muy engorrosa, ya que depende de la infraestructura de red y de la seguridad de la propia máquina. Por ejemplo, en muchos casos la máquina que analizamos puede tener un firewall (físico o lógico) que filtre el acceso entre tu máquina y la víctima, así que tendremos que afinar en nuestra búsqueda.

Existen distintos métodos para escanear los puertos, basándose en la forma de enviar tramas o en que protocolo centrarse. Todos estos métodos los veremos en el apartado práctico, así los iremos aplicando a medida que los vamos viendo.

La herramienta que vamos usar en este artículo es **Nmap**. Tal y como hemos visto en [host discovery](#), la herramienta de port scanning por excelencia es Nmap. En este caso no voy a dar otras herramientas ya que las demás son bastante más complejas y no tienen ni la mitad de funcionalidades que Nmap. Las virtudes principales que ofrece Nmap son:

- Multiplataforma
- Múltiple envío de paquetería de distintos protocolos
- Simple
- Ajuste de tiempo de escaneo para evitar sondeos

En este caso también existe un **exploit** para Metasploit Framework en `auxiliary/scanner/portscan/syn`. De todos modos, la propia web www.offensive-security.com recomienda el uso de nmap integrado en msf, por lo que no lo veremos.

Cabe destacar que Zenmap y las demás GUIs para Nmap NO son útiles tampoco en port scanning, así que trabajaremos sobre comandos continuamente.

Os presento unas herramientas de port scanning, para mi aún por mejorar, por si quereis echar un vistazo:

- Superscan (GUI demasiado simple para Windows, sin potencial)
- unicornscan (solo TCP)
- angryIPscanner (para mi la mejor tras nmap, solo para Windows y sin muchas funcionalidades)

Comando de nmap para port scanning:

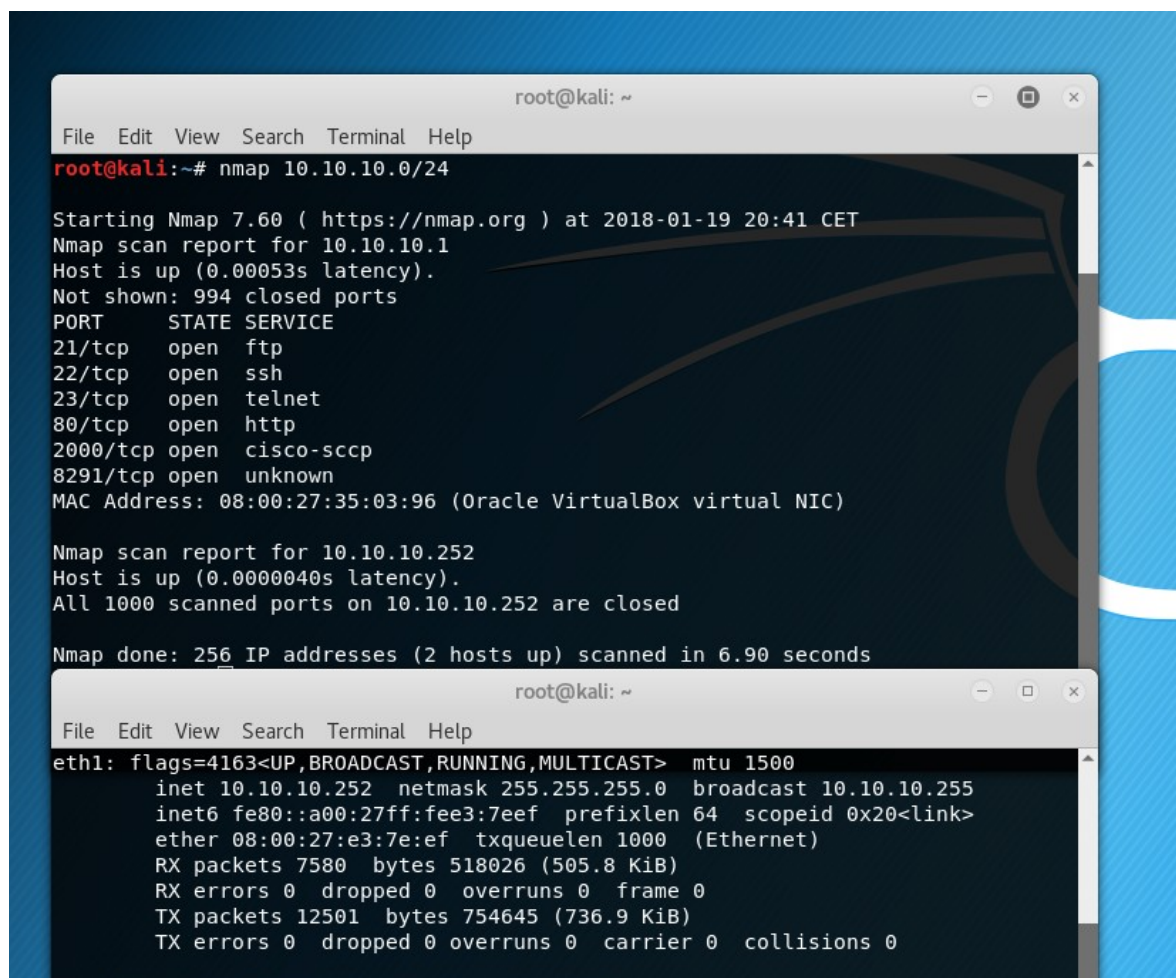
```
nmap -sX [ip_a_escanear] -p port1
```

Donde X es el tipo de escaneo

<https://mihackeo.blogspot.com.es/>

2. PASANDO DIRECTAMENTE A LA PRÁCTICA

Para hacer el port scanning vamos a ir poco a poco. En primer lugar tenemos que tener instalado Nmap y tener noción de hacer un [host discovery](#) (pulsando en el link te lleva a mi artículo de host discovery donde enseño a instalar Nmap y a hacer host discovery con el). Tras hacer host discovery en mi red obtengo esto:



```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# nmap 10.10.10.0/24  
  
Starting Nmap 7.60 ( https://nmap.org ) at 2018-01-19 20:41 CET  
Nmap scan report for 10.10.10.1  
Host is up (0.00053s latency).  
Not shown: 994 closed ports  
PORT      STATE SERVICE  
21/tcp    open  ftp  
22/tcp    open  ssh  
23/tcp    open  telnet  
80/tcp    open  http  
2000/tcp  open  cisco-sccp  
8291/tcp  open  unknown  
MAC Address: 08:00:27:35:03:96 (Oracle VirtualBox virtual NIC)  
  
Nmap scan report for 10.10.10.252  
Host is up (0.0000040s latency).  
All 1000 scanned ports on 10.10.10.252 are closed  
  
Nmap done: 256 IP addresses (2 hosts up) scanned in 6.90 seconds  
  
root@kali: ~  
File Edit View Search Terminal Help  
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 10.10.10.252 netmask 255.255.255.0 broadcast 10.10.10.255  
    inet6 fe80::a00:27ff:fee3:7eef prefixlen 64 scopeid 0x20<link>  
    ether 08:00:27:e3:7e:ef txqueuelen 1000 (Ethernet)  
    RX packets 7580 bytes 518026 (505.8 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 12501 bytes 754645 (736.9 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

La salida es un único host con IP 10.10.10.1, además de nuestra máquina. Como vemos, Nmap realiza el host discovery y, además, por cada máquina viva hace un pequeño port scanning. ESTE ESCaneo NO QUIERE DECIR QUE SÓLO TENGA ESOS PUERTOS ABIERTOS, aunque en este caso sí.

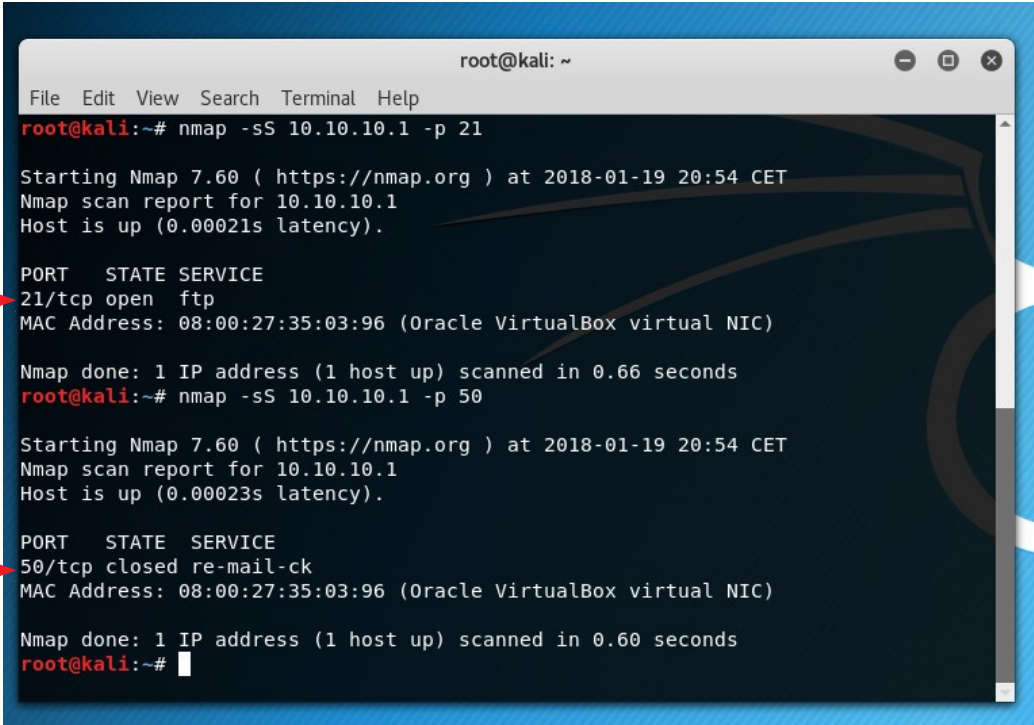
Ahora que sabemos que puertos hay abiertos y cuales no iremos técnica por técnica para ver si se cumple o no la teoría.

- TCP – SYN (-sS) : Es el que se realiza por defecto para conocer los puertos abiertos al hacer el host discovery. Se envía un paquete TCP con el flag SYNC activado, paquete usado para inicio de conexión.

RESPUESTA

- TCP SYNC-ACK → puerto abierto y permite conexión
- TCP RST → puerto cerrado
- No se recibe respuesta o se recibe ICMP-Unreacheable → filtrado (firewall habitualmente)

Probaremos con el puerto 21 (abierto) y puerto 50 (cerrado).



```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# nmap -sS 10.10.10.1 -p 21  
  
Starting Nmap 7.60 ( https://nmap.org ) at 2018-01-19 20:54 CET  
Nmap scan report for 10.10.10.1  
Host is up (0.00021s latency).  
  
PORT      STATE SERVICE  
21/tcp    open  ftp  
MAC Address: 08:00:27:35:03:96 (Oracle VirtualBox virtual NIC)  
  
Nmap done: 1 IP address (1 host up) scanned in 0.66 seconds  
root@kali:~# nmap -sS 10.10.10.1 -p 50  
  
Starting Nmap 7.60 ( https://nmap.org ) at 2018-01-19 20:54 CET  
Nmap scan report for 10.10.10.1  
Host is up (0.00023s latency).  
  
PORT      STATE SERVICE  
50/tcp    closed re-mail-ck  
MAC Address: 08:00:27:35:03:96 (Oracle VirtualBox virtual NIC)  
  
Nmap done: 1 IP address (1 host up) scanned in 0.60 seconds  
root@kali:~#
```

Vemos las respuestas (temas posteriores):

Source	Destination	Protocol	Length	Info
PcsCompu_e3:7e:ef	Broadcast	ARP	42	Who has 10.10.10.1? Tell 10.10.10.252
PcsCompu_35:03:96	PcsCompu_e3:7e:ef	ARP	60	10.10.10.1 is at 08:00:27:35:03:96
10.10.10.252	10.10.10.1	TCP	58	42341 → 21 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
10.10.10.1	10.10.10.252	TCP	60	21 → 42341 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1460
10.10.10.252	10.10.10.1	TCP	54	42341 → 21 [RST] Seq=1 Win=0 Len=0
10.10.10.252	10.10.10.1	TCP	58	42342 → 21 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
10.10.10.1	10.10.10.252	TCP	60	21 → 42342 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1460
10.10.10.252	10.10.10.1	TCP	54	42342 → 21 [RST] Seq=1 Win=0 Len=0
10.10.10.1	255.255.255.255	MNDP	162	53599 → 5678 Len=120
fe80::a00:27ff:fe35...	ff02::1	MNDP	182	5678 → 5678 Len=120
PcsCompu_35:03:96	CDP/VTP/DTP/PAGP/UD...	CDP	104	Device ID: MikroTik Port ID: ether2
PcsCompu_35:03:96	LLDP Multicast	LLDP	133	TTL = 120 System Name = MikroTik System Description = MikroTik
PcsCompu_e3:7e:ef	Broadcast	ARP	42	Who has 10.10.10.1? Tell 10.10.10.252
PcsCompu_35:03:96	PcsCompu_e3:7e:ef	ARP	60	10.10.10.1 is at 08:00:27:35:03:96
10.10.10.252	10.10.10.1	TCP	58	60764 → 50 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
10.10.10.1	10.10.10.252	TCP	60	50 → 60764 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
10.10.10.252	10.10.10.1	TCP	58	60765 → 50 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
10.10.10.1	10.10.10.252	TCP	60	50 → 60765 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Si analizamos la captura realizada mediante un sniffer, vemos nuestro envío SYN al 21 en primer lugar y al 50, y sus respectivos ACK y RST.

- **TCP CONNECT (-sT):** Realiza todo el **handshake** para ver si hay conexión total al puerto. Se suele realizar tras un TCP SYN exitoso para comprobar la total conexión a ese puerto.

Haremos de nuevo pruebas con el puerto 21 y 50, aunque al saber que está cerrado no es buena idea.

```

root@kali: ~
File Edit View Search Terminal Help

root@kali:~# nmap -sT 10.10.10.1 -p 21

Starting Nmap 7.60 ( https://nmap.org ) at 2018-01-19 21:08 CET
Nmap scan report for 10.10.10.1
Host is up (0.00025s latency).

PORT      STATE SERVICE
21/tcp    open  ftp
MAC Address: 08:00:27:35:03:96 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.48 seconds
root@kali:~# nmap -sT 10.10.10.1 -p 50

Starting Nmap 7.60 ( https://nmap.org ) at 2018-01-19 21:08 CET
Nmap scan report for 10.10.10.1
Host is up (0.00027s latency).

PORT      STATE SERVICE
50/tcp    closed re-mail-ck
MAC Address: 08:00:27:35:03:96 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.35 seconds
root@kali:~#

```

Y vemos que sucede cuando está abierto:

Source	Destination	Protocol	Length	Info
PcsCompu_e3:7e:ef	Broadcast	ARP	42	Who has 10.10.10.1? Tell 10.10.10.252
10.10.10.252	10.10.10.1	TCP	74	38790 → 21 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2648676406 TSecr=0 WS=128
10.10.10.1	10.10.10.252	TCP	74	21 → 38790 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=361236 TSecr=2648676406 WS=8
10.10.10.252	10.10.10.1	TCP	66	38790 → 21 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=2648676406 TSecr=361236
10.10.10.252	10.10.10.1	TCP	66	38790 → 21 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0 TSval=2648676406 TSecr=361236
10.10.10.1	255.255.255.255	MNDP	102	35399 → 5678 Len=120
fe80::a00:27ff:fe35...	ff02::1	MNDP	182	5678 → 5678 Len=120
PcsCompu_35:03:96	CDP/VTP/DTP/PAGP/UD...	CDP	104	Device ID: MikroTik Port ID: ether2
PcsCompu_35:03:96	LLDP_Multicast	LLDP	133	TTL = 120 System Name = MikroTik System Description = MikroTik RouterOS 6.39.3 (bugfix) x86
PcsCompu_e3:7e:ef	Broadcast	ARP	42	Who has 10.10.10.1? Tell 10.10.10.252
PcsCompu_35:03:96	PcsCompu_e3:7e:ef	ARP	60	10.10.10.1 is at 08:00:27:35:03:96
10.10.10.252	10.10.10.1	TCP	74	37254 → 50 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2648683091 TSecr=0 WS=128
10.10.10.1	10.10.10.252	TCP	60	50 → 37254 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Aquí vemos que nmap completa el handshake con su ACK y procede a hacer RST tras este. En el caso del puerto cerrado, no varía.

- **UDP (-sU):** Es un escaneo de puertos UDP. Era necesario incluir uno pero no es la mejor forma de hayar puertos UDP abiertos ya que existen 2 problemáticas:
 - Tendríamos que ir a ciegas probando puertos
 - Nunca podremos resolver la ambigüedad abierto-filtrado que se da a veces

RESPUESTAS:

- Paquete UDP → abierto
- Sin respuesta (muy habitual) → abierto o filtrado
- ICMP Unreachable Code=3 → cerrado
- ICMP Unreachable Code!=3 → filtrado

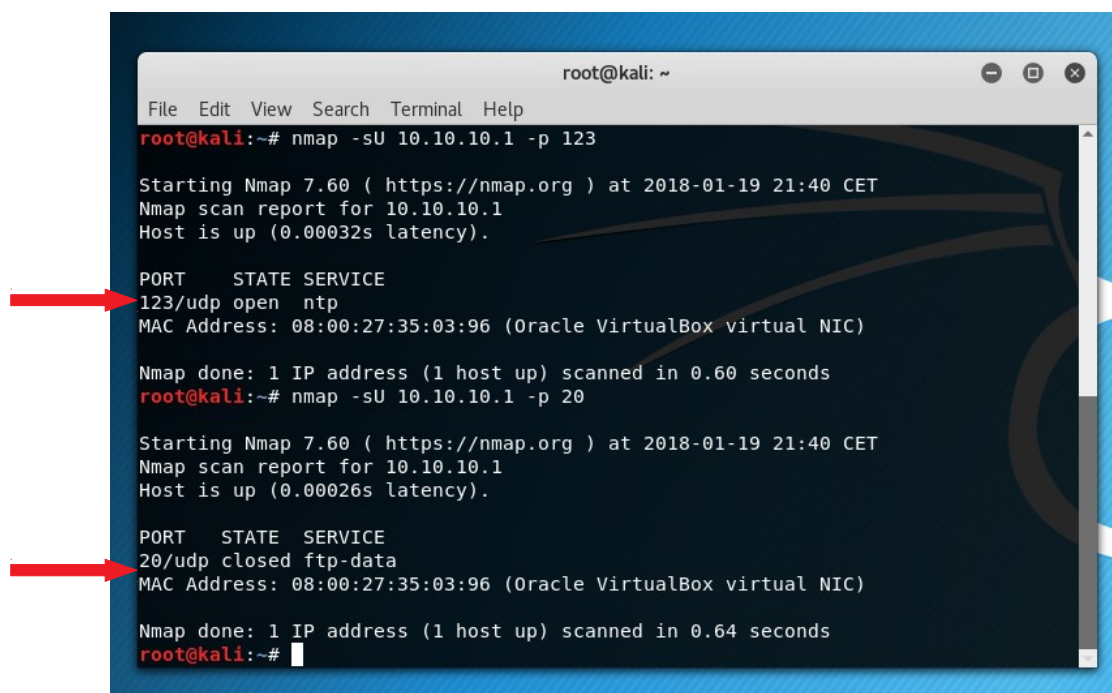
Muchas veces, lo más útil es escanear todos los puertos UDP.

Si queremos descubrir todos los puertos UDP (desde 1 a 1000) de golpe:

```
nmap -sU 10.10.10.1 -p 1-1000
```

En mi caso, el routerOS que estaba escaneando tenía los 1000 puertos cerrados, por tanto he abierto el de NTP (puerto 123). Para abrirlo solo es necesario acceder a **system > ntp > server** y, desde allí **set enabled=yes**

Ahora analizaremos entonces el puerto UDP 123 (abierto) y el 20 (cerrado):



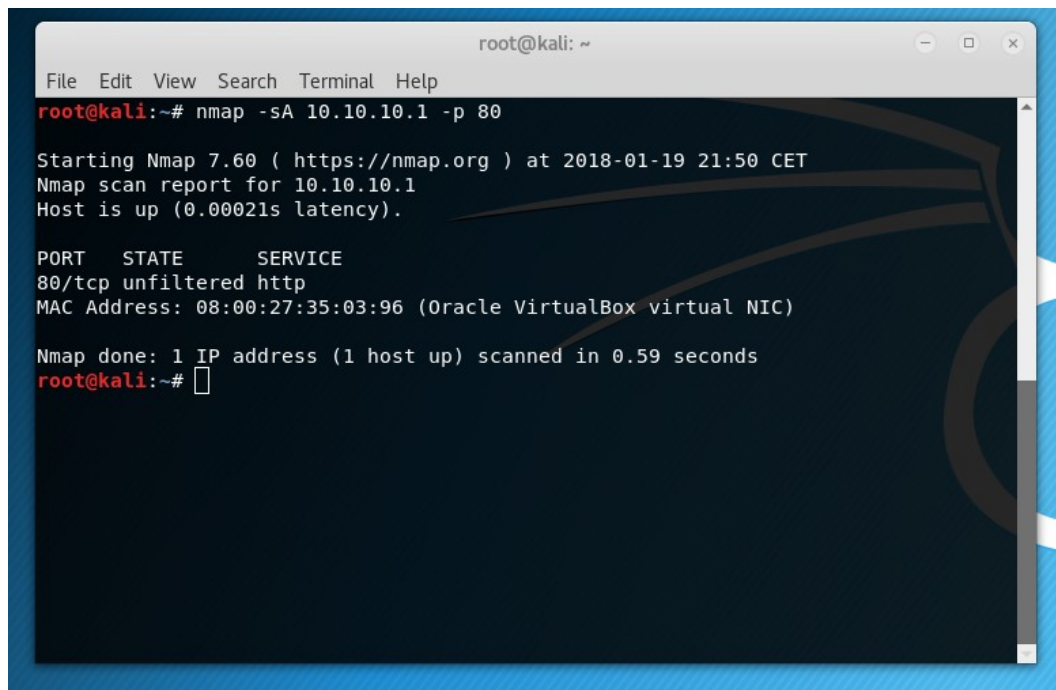
```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# nmap -sU 10.10.10.1 -p 123  
  
Starting Nmap 7.60 ( https://nmap.org ) at 2018-01-19 21:40 CET  
Nmap scan report for 10.10.10.1  
Host is up (0.00032s latency).  
  
PORT      STATE SERVICE  
123/udp   open  ntp  
MAC Address: 08:00:27:35:03:96 (Oracle VirtualBox virtual NIC)  
  
Nmap done: 1 IP address (1 host up) scanned in 0.60 seconds  
root@kali:~# nmap -sU 10.10.10.1 -p 20  
  
Starting Nmap 7.60 ( https://nmap.org ) at 2018-01-19 21:40 CET  
Nmap scan report for 10.10.10.1  
Host is up (0.00026s latency).  
  
PORT      STATE SERVICE  
20/udp    closed ftp-data  
MAC Address: 08:00:27:35:03:96 (Oracle VirtualBox virtual NIC)  
  
Nmap done: 1 IP address (1 host up) scanned in 0.64 seconds  
root@kali:~#
```

- **TCP ACK (-sA):** Envía un paquete TCP ACK. Es muy útil cuando nos filtran paquetes. Con esta técnica conoceremos si los firewalls son de estados o no y su comportamiento.

RESPUESTAS:

- TCP RST → No filtrado
- ICMP Unreachable o No respuesta → filtrado

En la red no hay firewalls, así que solo comprobaremos que un puerto (el 80 por ejemplo) no está filtrado.



```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# nmap -sA 10.10.10.1 -p 80  
  
Starting Nmap 7.60 ( https://nmap.org ) at 2018-01-19 21:50 CET  
Nmap scan report for 10.10.10.1  
Host is up (0.00021s latency).  
  
PORT      STATE      SERVICE  
80/tcp    unfiltered http  
MAC Address: 08:00:27:35:03:96 (Oracle VirtualBox virtual NIC)  
  
Nmap done: 1 IP address (1 host up) scanned in 0.59 seconds  
root@kali:~#
```

Nos marca unfiltered (no filtrado). Podemos ver también las tramas intercambiadas:

Source	Destination	Protocol	Length	Info
PcsCompu_e3:7e:ef	Broadcast	ARP	42	Who has 10.10.10.1? Tell 10.10.10.252
PcsCompu_35:03:96	PcsCompu_e3:7e:ef	ARP	60	10.10.10.1 is at 08:00:27:35:03:96
10.10.10.252	10.10.10.1	TCP	54	62022 → 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0
10.10.10.1	10.10.10.252	TCP	60	80 → 62022 [RST] Seq=1 Win=0 Len=0
10.10.10.252	10.10.10.1	TCP	54	62023 → 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0
10.10.10.1	10.10.10.252	TCP	60	80 → 62023 [RST] Seq=1 Win=0 Len=0
PcsCompu_35:03:96	PcsCompu_e3:7e:ef	ARP	60	Who has 10.10.10.252? Tell 10.10.10.1
PcsCompu_e3:7e:ef	PcsCompu_35:03:96	ARP	42	10.10.10.252 is at 08:00:27:e3:7e:ef

A esta técnica se asemeja la FIN (-sF) que marca el flag FIN, NULL que no marca ningún flag) y XMAS (que fija FIN, PSH y URG). Lo que sucede en este caso es que no se evalúa dicho paquete en el firewall y se deja pasar.

- **IDLE SCAN (-sI):** Muy sigiloso y permite conocer puertos de una máquina a través de un “zombie”. El zombie es una máquina ajena que hará un SYN-ACK en nuestro nombre aunque nosotros estemos totalmente filtrados en esa red.

El mecanismo es complejo y lento, pero efectivo. Primero se realiza un SYN-ACK con un FROM de una máquina Zombie a la víctima. Se comprueba el IP ID. Luego se realiza un SYN a la máquina zombie con el campo FROM con la IP de la víctima. La Zombie realiza un SYN-ACK a la máquina víctima y esta responde con un RST. Luego se procede a un SYN-ACK de nuestra máquina con FROM la del Zombie a la víctima y se comprueba de nuevo la IP ID. Si la nueva ID es la antigua +2 el puerto está abierto, si es ID + 1 está cerrado.

nmap -sI [ip_zombie] [ip_victima]

Existen mas escaneos, pero con estes deberían llegar para conocer los puertos de cualquier máquina atacable. Si quieres conocer más: <http://anish.at.preempted.net/nmap.htm>

3. CONCLUSIONES

Para conocer bien a la víctima, es necesario conocer su equipo y a que paquetería es susceptible por cada puerto abierto. Para ello indiscutiblemente nmap es la mejor opción ya que con unos simples comandos puedes llegar a tener un conocimiento de equipos y sus respectivos puertos muy grande.

Tras conocer esto, el próximo paso es conocer el interior del equipo, su Sistema Operativo, sus servicios... A este mecanismo se le denomina fingerprinting.