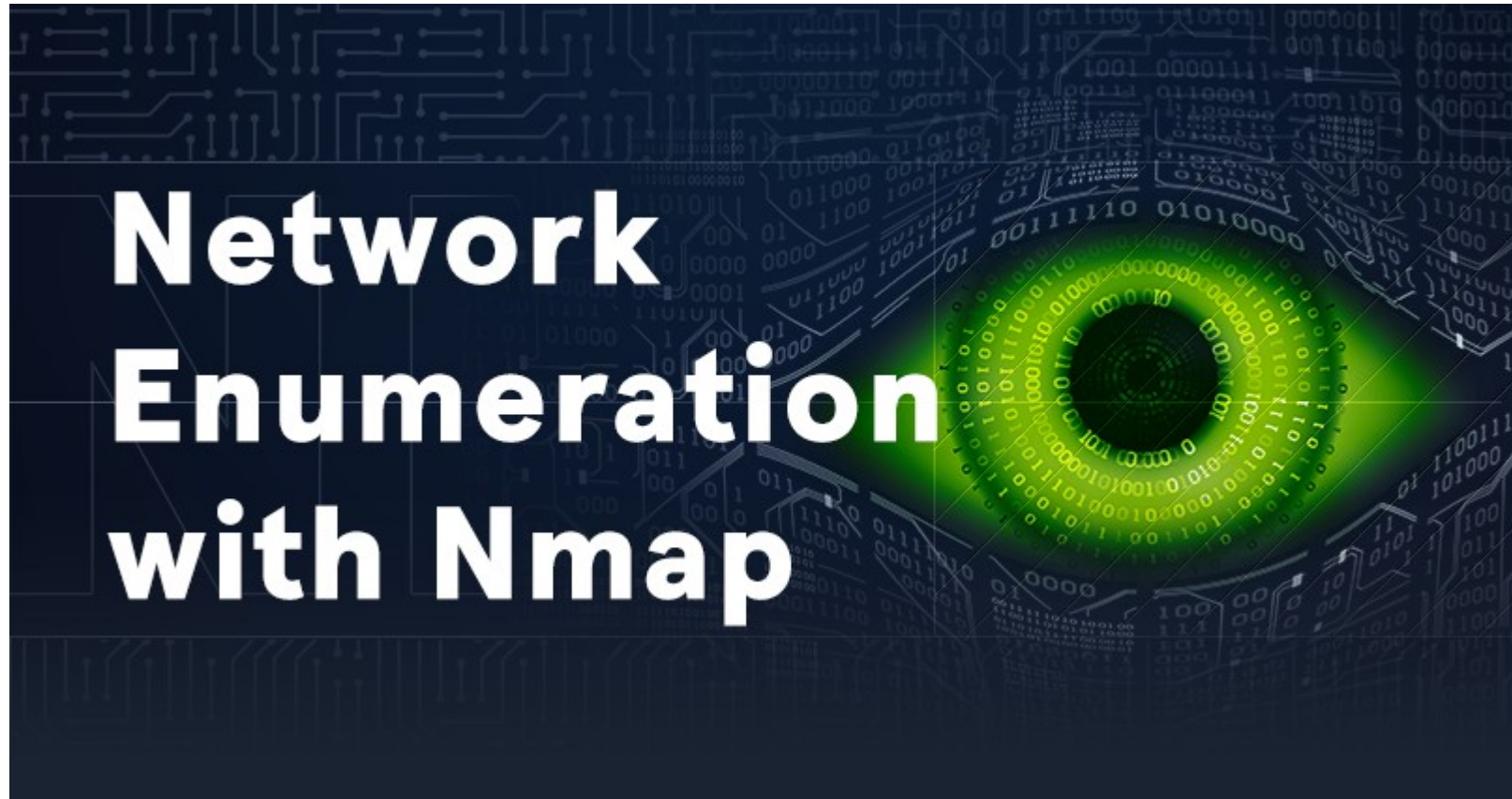


# ESCANEEO Y ENUMERACIÓN DE REDES Y SERVICIOS



# Índice de contenido

1. Introducción.
2. Técnicas de escaneo.
3. Nmap.
  1. Host discovery con nmap.
  2. Service discovery (scan de puertos) con nmap.
  3. Otras utilidades.
  4. Webmap.
4. Detección y evasión de firewalls.
5. Scripts NSE (Nmap Scripting Engine).
6. Enumeración con Metasploit.
7. Análisis automático de vulnerabilidades (OpenVAS, Nessus, Legion).

# Introducción

## Fases del Hacking

### ► Para un **cracker**

- Footprinting
- **Fingerprinting**
- Acceso o Explotación
- Pivoting/Escalada de privilegios
- Mantener acceso
- Borrar huellas

### ► Para un **hacker ético**

- Footprinting
- **Fingerprinting**
- Acceso o Explotación
- Pivoting/Escalada de privilegios
- Escribir informe
- Presentar informe

# Introducción

## Fingerprinting (Escaneo y Enumeración)

### ► Fase de **escaneo + enumeración**

- **Escaneo**: identificar host activos + servicios en cada host.
- **Enumeración**: profundizar en cada servicio de cada host.

### ► **Objetivo**: recopilar toda la información posible de host activos y versiones de los servicios, aplicaciones web, protocolos, que están ejecutándose, para encontrar brechas de seguridad.

# Introducción

Hasta esta fase ...

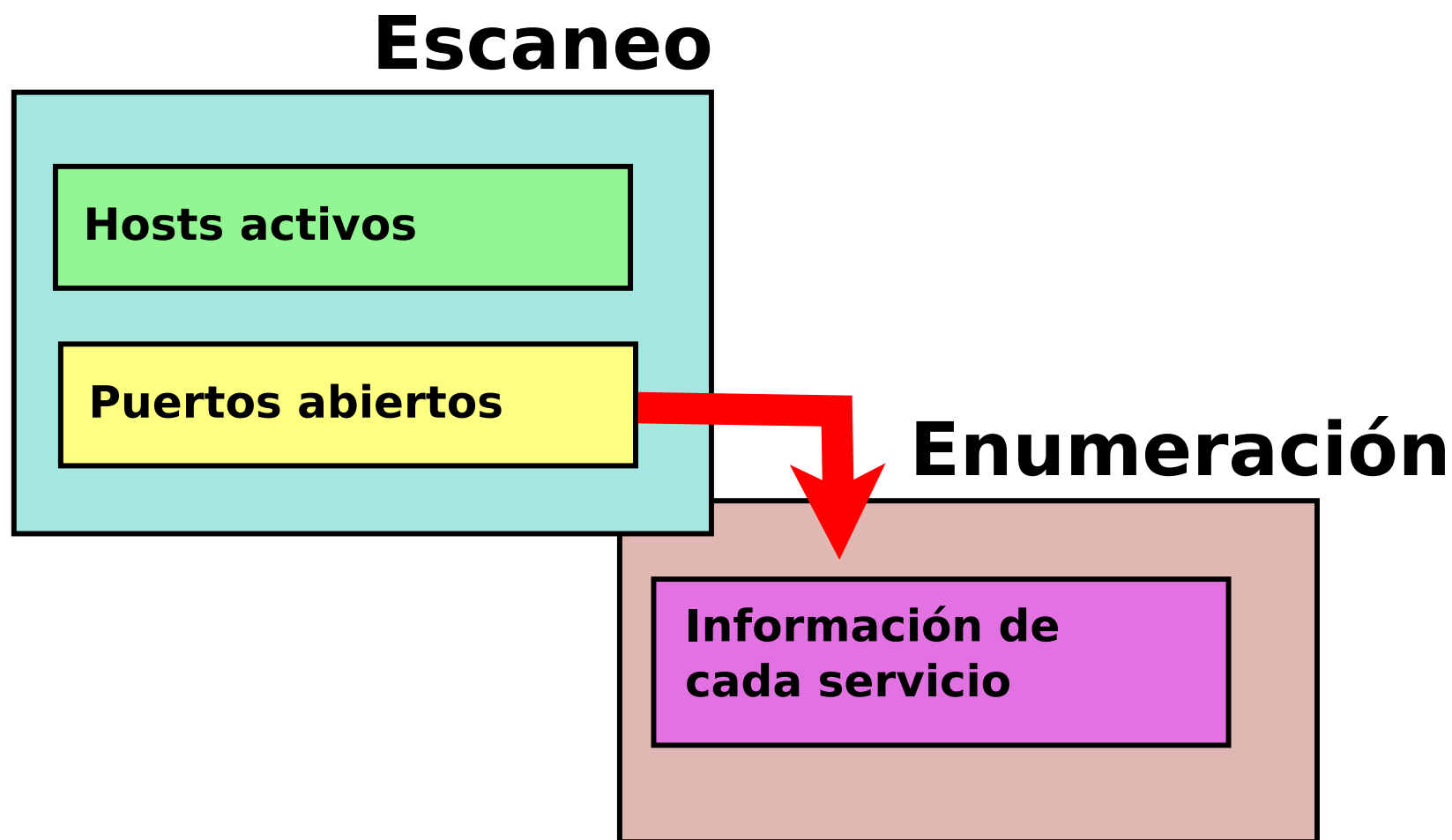
- ▶ Tras **footprinting** (fase 1), tenemos:
  - Hacking externo: IPs públicas, dominios, ...
  - Hacking interno: subredes, organización

En esta fase ...

- ▶ El **objetivo** es:
  - Identificar hosts activos: IPs, MAC, SO, nombres, ...
  - Puertos abiertos en cada host.
  - Identificar vulnerabilidades en los servicios encontrados.
  - Localizar recursos accesibles que no están debidamente protegidos

# Introducción

Relación entre las fases de escaneo y enumeración



# Introducción

## Herramientas que usaremos

- ▶ **Nmap** (<https://nmap.org/>). Una de las herramientas más potentes y que debemos conocer en profundidad.
- ▶ **Rustscan** (<https://rustscan.github.io/RustScan/>). Escáner moderno desarrollado en Rust que escanea los 65k puertos en menos de 10 segundos.
- ▶ **Masscan** (<https://github.com/robertdavidgraham/masscan>). Escáner enfocado a escaneo de grandes segmentos de redes. Sus creadores afirman que pueden escanear Internet en menos de 5 minutos.
- ▶ **Metasploit** (<https://www.metasploit.com/>). Dispone de numerosos módulos auxiliares para realizar escaneo y enumeración.
- ▶ **Legion** (<https://govanguard.com/legion/>). Herramienta automática de escaneo, enumeración y análisis de vulnerabilidades. Se basa en la antigua herramienta Sparta.
- ▶ **Nessus** (<https://es-la.tenable.com/products/nessus>). Escáner automático de vulnerabilidades propietaria de Tenable. Dispone de versión libre para uso académico.
- ▶ **OpenVAS** (<https://www.openvas.org/>). Escáner automático de vulnerabilidades de código abierto.

# Técnicas de escaneo

- ▶ Recuerda que en la fase de escaneo, teníamos 2 subfases:
  - 1) Descubrimiento de hosts
  - 2) Identificación de puertos activos por host



# Técnicas de escaneo

## Descubrimiento de hosts

- ▶ Barrido para identificar host activos dentro de un rango de IPs.
- ▶ **Técnicas:**
  - Barrido Ping.
  - TCP SYN ping.
  - TCP ACK ping.
  - UDP ping.
  - ARP ping.

# Técnicas de escaneo

## Descubrimiento de servicios

- ▶ Se hace mediante el envío de sondas (TCP o UDP) a un puerto y observar la respuesta.
- ▶ **Posibles estados** (nomenclatura nmap)
  - **Abierto**: servicio escuchando.
  - **Cerrado**: no hay servicio.
  - **Filtrado**: puerto no accesible.
- ▶ **Casos especiales**
  - **No-filtrado, abierto-filtrado, cerrado-filtrado.**

# Técnicas de escaneo

## Descubrimiento de servicios - técnicas

- ▶ Escáner SYN o Half-Open.
- ▶ Escáner Full Connect.
- ▶ Escáner UDP.
- ▶ Escáner ACK.
- ▶ Escáneres especiales:
  - Null-scan.
  - Fin-scan.
  - XMAS-Scan.

**Los veremos  
practicando  
con nmap!**

# nmap

- ▶ Network **map**per.
- ▶ Herramienta **Open source** y **multiplataforma**
- ▶ Usado para escanear redes en auditorías de seguridad.
- ▶ Herramienta básica de todo pentester:
  - Proporciona información de gran valor para etapas posteriores del pentesting.

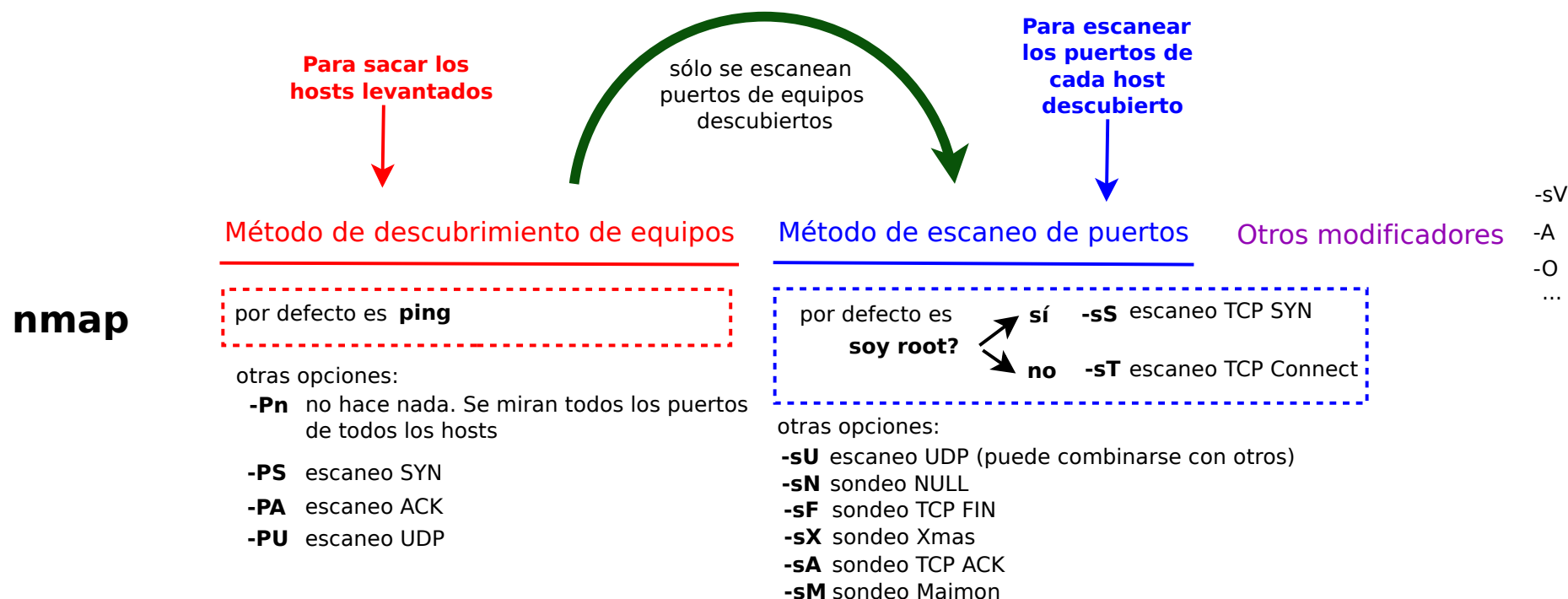
# nmap

## Funciones de nmap

- ▶ **Mapeo** de redes y **descubrimiento** de equipos.
- ▶ Escáner de **puertos**.
- ▶ Detección de **sistemas operativos**.
- ▶ Detección de **versiones de servicios**.
- ▶ **Enumeración** de servicios.
- ▶ **Detección y evasión** de firewalls.
- ▶ ...

# nmap

## Estructura del comando



**Pablo Esteban Sánchez**  
pablo.profe.tic@gmail.com

<https://nmap.org/book/toc.html>

<https://www.hackingarticles.in/category/nmap/>

# nmap

- ▶ Importante tener en cuenta la diferencia entre ejecutar nmap:
  - como **root** (o con sudo) o como **Administrador** en Windows → capacidad de forjar segmentos
  - vs
  - ejecutarlo como **usuario normal** → llamadas al sistema
- ▶ **Conclusión:** ejecuta nmap como root (o con sudo)

# Host discovery con nmap

## HOSTS DISCOVERY:

- sL:** List Scan - simply list targets to scan
- sn:** Ping Scan - disable port scan
- Pn:** Treat all hosts as online -- skip host discovery
- PS/PA/PU/PY[portlist]:** TCP SYN/ACK, UDP or SCTP discovery to given ports
- PE/PP/PM:** ICMP echo, timestamp, and netmask request discovery probes
- PO[protocol list]:** IP Protocol Ping
- n/-R:** Never do DNS resolution/Always resolve [default: sometimes]
- dns-servers <serv1[,serv2],...>:** Specify custom DNS servers
- system-dns:** Use OS's DNS resolver
- traceroute:** Trace hop path to each host

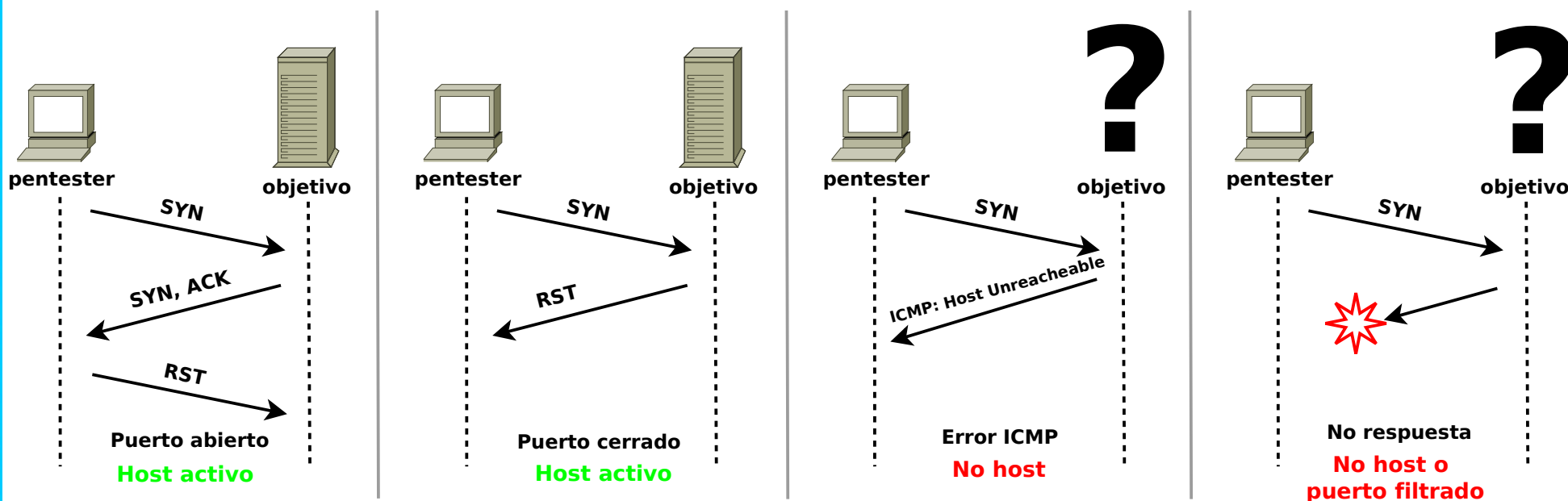
<https://www.hackingarticles.in/nmap-for-pentester-host-discovery/>



# Host discovery con nmap

## TCP SYN ping (-PS)

- ▶ Envía segmento TCP con flag **SYN** activo
  - Simula inicio de conexión **Three Way-handshake**

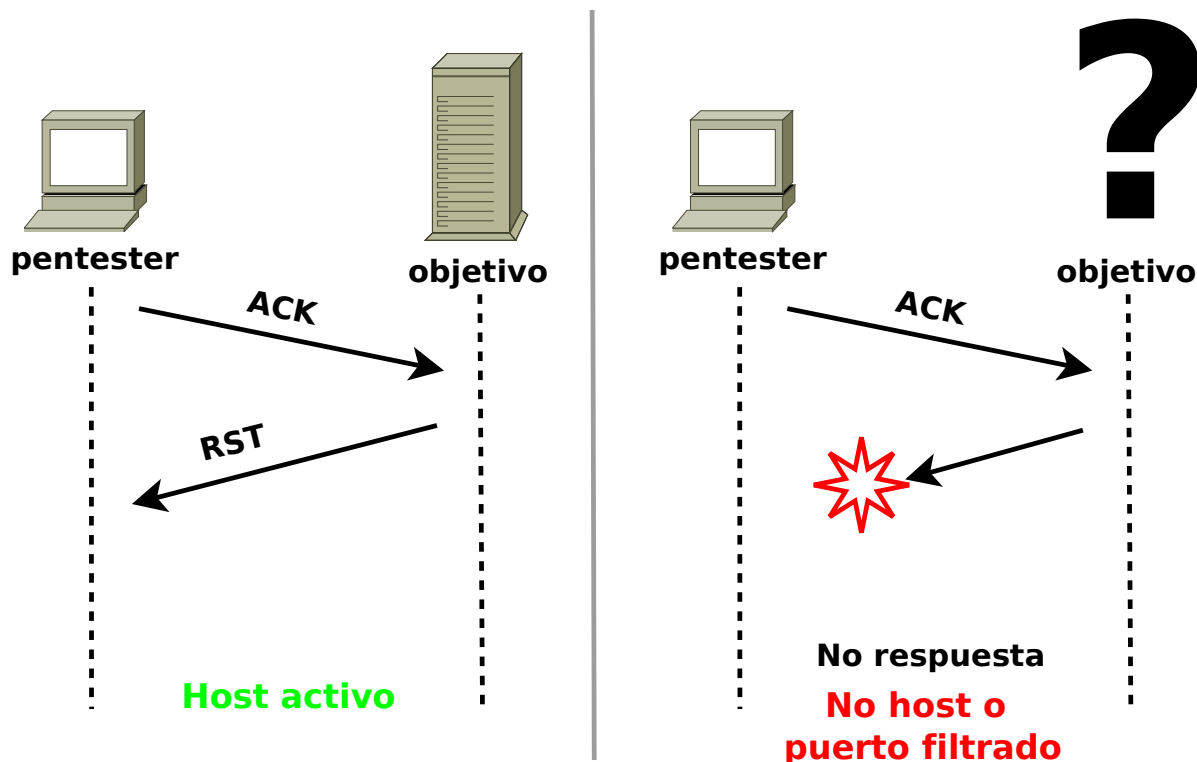


**La conexión se termina abruptamente**

# Host discovery con nmap

## TCP ACK ping (-PA)

- ▶ Envía segmento TCP con flag **ACK** activo
  - Más probabilidad de evadir cortafuegos!



# Host discovery con nmap

## UDP ping (-PU)

- ▶ Envía datagrama UDP en lugar de TCP
  - Evade configuraciones de FW que se olvidan de UDP.
- ▶ **Respuesta:**
  - ICMP Port Unreachable → máquina en pie
  - Otro error ICMP o TTL agotado → no hay host

# Host discovery con nmap

## ARP ping (-PR)

- ▶ Especialmente indicado en redes locales
- ▶ Es rápido y eficiente
- ▶ De hecho, es el usado por defecto cuando atacante y destinos están en la misma red
  - Para evitar este comportamiento, incluir opción  
`--send-ip`

# Host discovery con nmap

## IP ping (-PO)

- ▶ Permite realizar análisis eligiendo el/los protocolos que corre sobre IP.

`-PO[protocol list]`

- ▶ Los protocolos se indican en formato IANA (valor numérico)

- <https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml>

- ▶ Por ejemplo, permite en una sola ejecución, probar un puerto TCP y un UDP.

# Host discovery con nmap

## Otros métodos

- ▶ Omitir descubrimiento de equipos (**-Pn**)
  - Si sabemos 100% que los hosts destino están levantados
- ▶ ICMP Echo Ping (**-PE**)
  - Usando Echo requests
  - Equivalente a usar el comando ping
- ▶ ICMP Timestamp (**-PP**)
  - Usa mensajes ICMP timestamp
  - Útil contra FW que bloquean pings

# Host discovery con nmap

## Técnica de descubrimiento por defecto

- ▶ Para usuarios **con privilegios** se realiza un:
  - -PS443,80
  - -PA80
- ▶ Para usuarios **sin privilegios**:
  - -PS80,443

# Host discovery con nmap

Antes de pasar al escáner de puertos ...

## ▶ **Traceroute** con nmap

- Complemento a cualquier análisis añadiendo opción `--traceroute`

## ▶ **Reverse address resolution**

- IP → nombres de dominio
- Complemento a cualquier análisis añadiendo opción `-R`



# Análisis de puertos

- ▶ Recuerda lo que hace nmap cuando lo ejecutas:
  - 1) Identifica **equipos activos** en el rango suministrado
  - 2) Para los equipos identificados en 1) analiza **qué puertos están abiertos**
- ▶ En este apartado nos centramos en 2)
- ▶ Por defecto, se analizan los 1000 top-ports tcp
  - # `nmap --top-ports 1000 -d -oG - localhost`

<https://nullsec.us/top-1-000-tcp-and-udp-ports-nmap-default/>

<https://www.hackingarticles.in/network-scanning-using-nmap-beginner-guide/>

# Análisis de puertos

- Hay distintas técnicas basadas en la manipulación de flags.

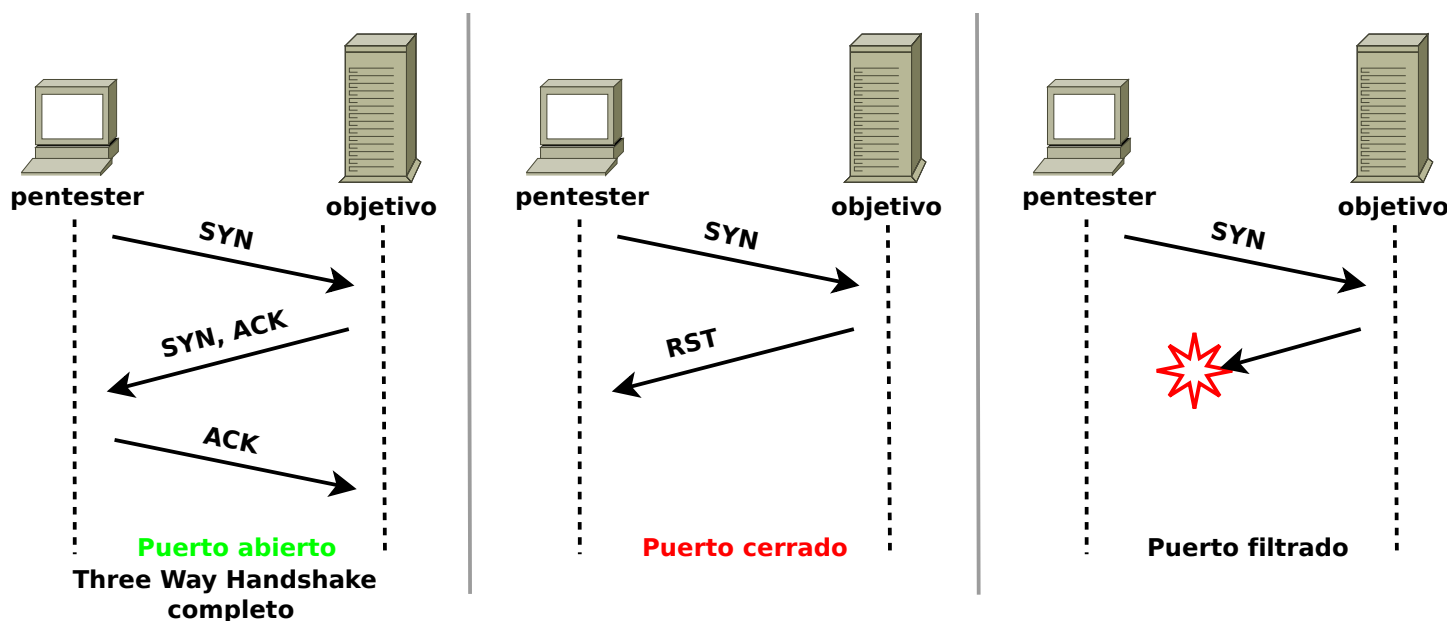
## SCAN TECHNIQUES:

- sS/sT/sA/sW/sM:** TCP SYN/Connect()/ACK/Window/Maimon scans
- sU:** UDP Scan
- sN/sF/sX:** TCP Null, FIN, and Xmas scans
- scanflags <flags>:** Customize TCP scan flags
- sI <zombie host[:probeport]>:** Idle scan
- sY/sZ:** SCTP INIT/COOKIE-ECHO scans
- sO:** IP protocol scan
- b <FTP relay host>:** FTP bounce scan

# Análisis de puertos

## TCP Connect Scan (-sT)

- ▶ También llamado **Full Open Scan**.
- ▶ Trata de realizar **conexión completa** con el objetivo (**Three Way Handshake**)



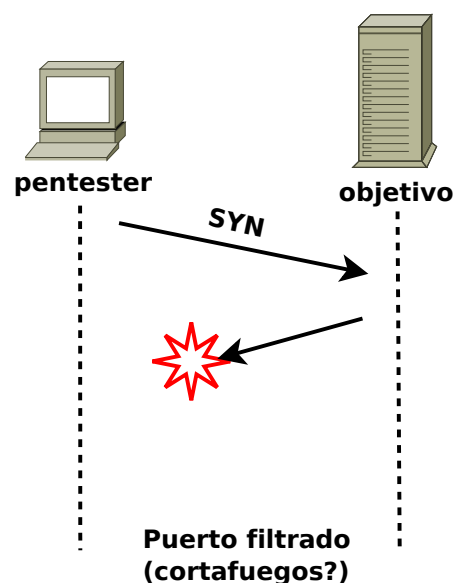
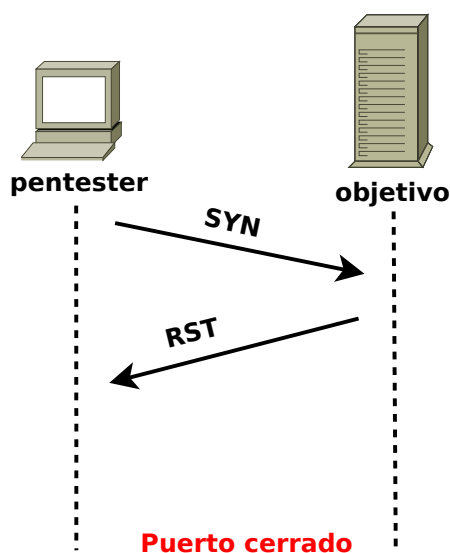
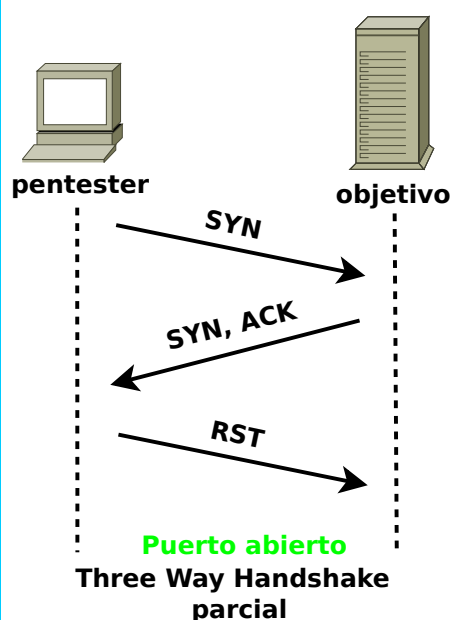
### Inconvenientes

- Lento
- Poco sigiloso

# Análisis de puertos

## TCP SYN Scan (-sS)

- ▶ También llamado **Full Open Scan**.
- ▶ Trata de realizar **conexión completa** con el objetivo (**Three Way Handshake**)



### Ventajas

- Rápido
- Eficiente
- Más sigiloso
- Bypass IDS

# Análisis de puertos

## UDP Scan (-sU)

- ▶ Los métodos vistos hasta ahora son para **TCP**
- ▶ Este permite escaneo de puertos **UDP**.
- ▶ Recuerda que UDP es no orientado a conexión:
  - No hay **Three Way handshake**
  - **Datagramas más simples**: no flags, no fragmentos, ...

# Análisis de puertos

## UDP Scan (-sU)

- ▶ Respuesta más complicada de interpretar:
  - Si recibes **ICMP** (Port Unreachable) → puerto **cerrado**
  - **No respuesta** → puerto **abierto** u open | filtered
  - **Respuesta** → **puerto abierto**

# Análisis de puertos

## Escáneres especiales

- ▶ Útiles para evadir FW que filtran flag SYN
- ▶ Funcionan bien contra Linux y BSD, no tanto con Windows
  - Fin Scan (**-sF**)
    - Segmento con flag **FIN** activo
  - XMAS Scan (**-sX**)
    - Segmento con flags **FIN**, **URG** y **PUSH** activos
  - Null Scan (**-sN**)
    - Segmento con todos los flags inactivos

# Análisis de puertos

## Escáneres especiales

### ► Interpretación Respuestas:

- No respuesta: puerto open | filtered
- Segmento con RST activo: puerto **cerrado**
- ICMP unreachable (tipo 3, código 1, 2, 3, 9, 10 ó 13): puerto filtrado



# Utilidades Escáner de Puertos


## Detección del sistema operativo (-o)

- ▶ Podemos añadirlo a cualquier análisis
- ▶ Nos da un % de probabilidad del SO del objetivo.
- ▶ La información que muestra es la siguiente:
  - **Device type:** indica el tipo de dispositivo, como *router*, *printer*, *firewall*, *general purpose*...
  - **Running:** Muestra la familia del SO y la generación si es posible.
  - **OS CPE:** Muestra el *Common Platform Enumeration* del SO. Comienza con **cpe/o** (SO) o **cpe/h** (hardware).
  - **OS Details:** Muestra una descripción más extensa en formato más legible.

# Utilidades Escáner de Puertos

## Detección de servicios (-sV)

- ▶ Podemos añadirlo a cualquier análisis.
- ▶ Intenta sacar el software concreto y la versión del servicio asociado al puerto abierto
- ▶ Muy útil para buscar vulnerabilidades.
  - <https://cvedetails.com>
  - <https://www.exploit-db.com/>

```
root@kali:~# nmap -sV 192.168.1.127   
  
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-17 06:53 EST  
Nmap scan report for 192.168.1.127  
Host is up (0.0020s latency).  
Not shown: 989 closed ports  
PORT      STATE SERVICE      VERSION  
80/tcp    open  http         Microsoft IIS httpd 7.5  
135/tcp   open  msrpc        Microsoft Windows RPC  
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn  
445/tcp   open  microsoft-ds Microsoft Windows 7 - 10 microsoft-ds (workgroup: WORKGROUP)  
5000/tcp   open  tcpwrapped  
49152/tcp open  msrpc        Microsoft Windows RPC  
49153/tcp open  msrpc        Microsoft Windows RPC  
49154/tcp open  msrpc        Microsoft Windows RPC  
49155/tcp open  msrpc        Microsoft Windows RPC  
49156/tcp open  msrpc        Microsoft Windows RPC  
49157/tcp open  msrpc        Microsoft Windows RPC  
MAC Address: E0:14:31:00:00:07:AA (Apple)  
Service Info: Host: WIN-EHVJ41TLTLA; OS: Windows; CPE: cpe:/o:microsoft:windows
```

# Utilidades Escáner de Puertos

## Detección de Versiones de servicios (-sV)

### ► Parámetros adicionales:

- **--version-intensity N** [N entre 0 y 9][0 light, 9 agresivo]
  - **Ej:** `nmap -sV -version-intensity 8 -p 80 192.168.1.1`
- **--version-light** equivale a `--version-intensity 0`
- **--version-all** equivale a `--version-intensity 9`

#### RECOMENDABLE en la práctica:

Usa una **detección normal** o **ligera** para todos los puertos

Usa una **detección agresiva** contra un puerto individual

#### **DETECCIÓN AGRESIVA:**

**PROS:** > probabilidad de éxito

**CONTRAS:** Requiere mucho tiempo, llama la atención, puede saturar red (posible DOS)

# Utilidades Escáner de Puertos

## Aggressive Scanning (-A)

### ► Combina **3 escáneres**:

- Detección del **sistema operativo** (-O)
- Detección de **servicios** (-sV)
- Ejecución de algunos **scripts de enumeración** (-sC).

### ► ¿**Scripts de enumeración**?

- Lo veremos en el apartado de enumeración.
- Nmap incorpora sistema modularizable de scripts.
- Entre otras cosas permiten recabar información adicional de los servicios.

# Utilidades Escáner de Puertos

## Fast Scan (-F)

- ▶ Escáner de puertos rápido.
- ▶ Mira el estado de los 100 puertos más comunes.
  - Útil para hacerse una idea general de la situación
- ▶ Mira el primer apartado en `/etc/services`

# Guardar la salida de nmap

- ▶ Aunque nmap ofrece muchas posibilidades para guardar informes, nos centramos en:

OUTPUT:

```
-oN/-oX/-oS/-oG <file>: Output scan in normal, XML, s|<rIpt kIddi3,  
and Grepable format, respectively, to the given filename.  
-oA <basename>: Output in the three major formats at once
```

- ▶ Básicamente:

- oN: salida normal de nmap (\*.nmap)
- oX: salida en XML (\*.xml)
- oG: salida en formato grepable (\*.gnmap)
- oA: guarda en los 3 formatos anteriores

# WebMap

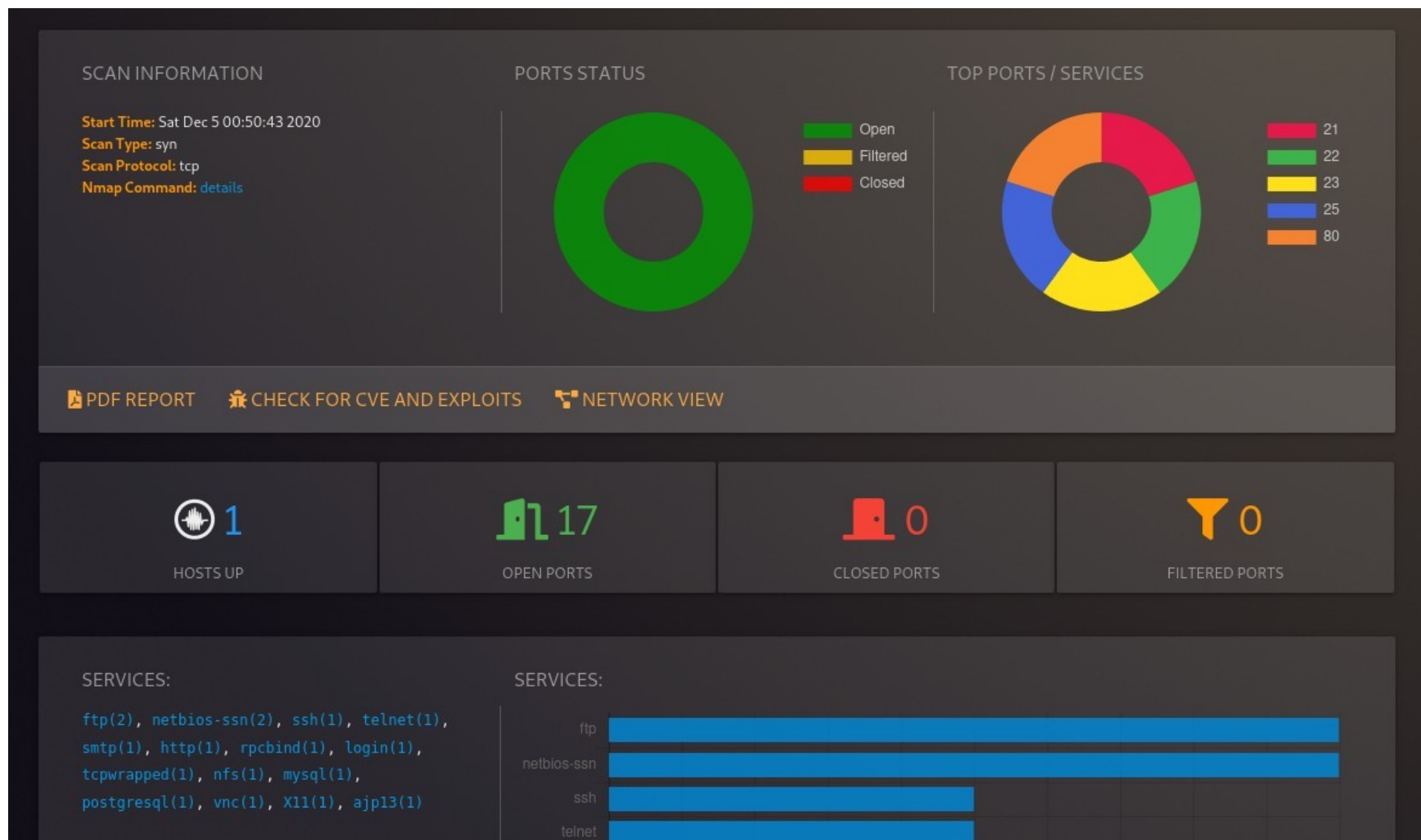
## Un Dashboard para nmap

- ▶ Dashboard para nmap.
- ▶ Multitud de posibilidades de reportes.
- ▶ Puede lanzar escáneres e importar resultados.
- ▶ Genera informes visualmente atractivos.
- ▶ Recomendable usarlo con docker.

<https://github.com/SabyasachiRana/WebMap>



# WebMap





# WebMap

## NSE Scripts for 10.0.2.4:

**ftp-anon** - Address: **10.0.2.4** - Port: **21**

```
Anonymous FTP login allowed (FTP code 230)
```

**ftp-syst** - Address: **10.0.2.4** - Port: **21**

```
STAT:  
FTP server status:  
Connected to 10.0.2.20  
Logged in as ftp  
TYPE: ASCII  
No session bandwidth limit  
Session timeout in seconds is 300  
Control connection is plain text  
Data connections will be plain text  
vsFTPD 2.3.4 - secure, fast, stable  
End of status
```

**ssh-hostkey** - Address: **10.0.2.4** - Port: **22**

```
1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)  
2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
```

**smtp-commands** - Address: **10.0.2.4** - Port: **25**

```
metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME,  
DSN,
```

# Detección y evasión de FW

- ▶ Nmap incorpora utilidades para:
  - Confirmar existencia de cortafuegos en un objetivo.
  - Traspasarlos con cierto grado de garantía.

## EVASIÓN Y FALSIFICACIÓN PARA CORTAFUEGOS/IDS:

```
-f; --mtu <valor>: fragmentar paquetes (opc. con el MTU indicado)
-D <señuelo1,señuelo2[,ME],...>: Disimular el análisis con señuelos
    N. del T.: «ME» es «YO» mismo.
-S <Dirección_IP>: Falsificar la dirección IP origen
-e <interfaz>: Utilizar la interfaz indicada
-g/--source-port <numpuerto>: Utilizar el número de puerto dado
--data-length <num>: Agregar datos al azar a los paquetes enviados
--ttl <val>: Fijar el valor del campo time-to-live (TTL) de IP
--spoof-mac <dirección mac/prefijo/nombre de fabricante>: Falsificar la dirección
MAC
--badsum: Enviar paquetes con una suma de comprobación TCP/UDP falsa
```

# Detección y evasión de FW

## Detección de FW con sondas ACK (-sA)

- ▶ Envío de segmentos con flag ACK activo
- ▶ Se interpreta la respuesta:
  - **No respuesta** → posibilidad de puerto filtrado
  - **RST** (segmento con flag RST activo) → puerto no filtrado (OJO!! puede estar abierto o no!!)
- ▶ Recuerda: estas pruebas es sólo para ver si hay filtrado de puertos.

<https://www.hackingarticles.in/understanding-guide-nmap-firewall-scan-part-1/>

<https://www.hackingarticles.in/understanding-guide-nmap-firewall-scan-part-2/>

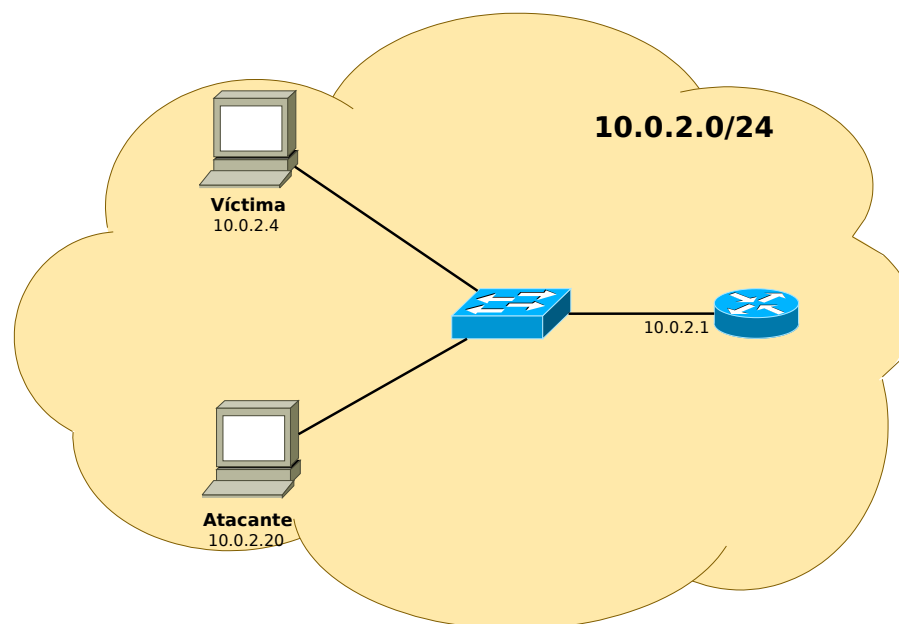
# Detección y evasión de FW

## Uso de decoy IP (-D)

- ▶ Cortina de humo para engañar a posibles IDS
- ▶ Consiste en indicar qué IPs aparecerán como IP origen en los paquetes sonda que envíes
  - Echar la culpa a otros...

# Detección y evasión de FW

## Uso de decoy IP (-D)



- ▶ Desde la 10.0.2.20 podemos lanzar sondas a 10.0.2.4 indicando que su origen es 10.0.2.1 (una IP legítima)

Source	Destination	Protocol	Length	Info
10.0.2.20	10.0.2.4	TCP	58	56654 → 80 [SYN] Seq=0
10.0.2.1	10.0.2.4	TCP	58	56654 → 80 [SYN] Seq=0
10.0.2.4	10.0.2.20	TCP	60	80 → 56654 [SYN, ACK]
10.0.2.20	10.0.2.4	TCP	54	56654 → 80 [RST] Seq=1

# Detección y evasión de FW

## MAC Spoofing

- ▶ Cambiar de manera dinámica nuestra MAC
- ▶ Se hace con **--spoof-mac [MAC]**
- ▶ Permite generar MACs de fabricantes:
  - **--spoof-mac Dell**
- ▶ ¡Sólo tiene sentido si atacante y víctima están en la misma red local!

# Enumeración

## ► Recuerda ...

### – Escaneo:

- Identificar **equipos activos**
- Identificar **puertos abiertos** en equipos activos

### – Enumeración:

- Obtener **información adicional** a partir de los puertos abiertos: listas de usuarios, default passwords, ...

## ► Hay muchos protocolos susceptibles de enumeración:

- Debido a fallas de programación.
- Por mala configuración.

# Enumeración con nmap

## NSE (Nmap Scripting Engine)

- ▶ Nmap permite enumeración a través de **NSE**.
- ▶ **NSE**: Nmap Scripting Engine.
  - **Framework** para la elaboración de scripts
  - Scripts escritos en **LUA**
  - Fácilmente **integrables en nmap**
  - Hay un gran número preinstalados en */usr/share/nmap/scripts*

<https://nmap.org/book/nse.html>



# Enumeración con nmap

## NSE (Nmap Scripting Engine)

- ▶ Se puede consultar la lista completa de preinstalados en <https://nmap.org/nsedoc/index.html>
- ▶ Permiten, entre otras cosas:
  - Detección de redes
  - Detección de versiones más sofisticada
  - Detección de vulnerabilidades
  - Detección de backdoors
  - ...

# Enumeración con nmap

## NSE (Nmap Scripting Engine)

► Se organizan en **categorías**:

- Una categoría incluye varios scripts
- Un scripts puede pertenecer a varias categorías

► Ejemplos de categorías:

- **default**: Los scripts en esta categoría se ejecutan con las opciones -sC o -A.
- **auth**: Tratan de encontrar credenciales en el objetivo (distinto de ataque por fuerza bruta. *ftp-anon, oracle-enum-users, ...*).
- **brute**: Intenta adivinar credenciales mediante fuerza bruta (*http-brute, oracle-brute, snmp-brute, ...*).
- **dos**: Realizan ataques de denegación de servicio.
- **intrusive**: Scripts que no se pueden categorizar como *safe* puesto que pueden causar que el objetivo deje de funcionar.
- **discovery**: Trata de descubrir más información de la red o del servicio (*html-title, smb-enum-shares, ...*).
- **exploit**: Tratan de explotar de forma activa alguna vulnerabilidad (*http-shellshock*).
- **vuln**: Tratan de reportar vulnerabilidades conocidas.

# Enumeración con nmap

## NSE (Nmap Scripting Engine)

- ▶ Se lanzan al mismo tiempo que un análisis normal.
- ▶ Puedes ejecutar **1 script** o una **categoría completa**
- ▶ Uso del parámetro `--script` y para pasar parámetros `--script-args`
- ▶ Ej:

```
- nmap -p 21 --script ftp-anon ftp.rediris.es
- nmap --script http-methods --script-args http-methods.test-all=true scanme.nmap.org
- nmap --script "http-*"
- nmap --script "not intrusive"
- nmap --script "default or safe"
- nmap --script "default and safe"
- Nmap --script "(default or safe or intrusive) and not http-*
```

<https://nmap.org/book/man-nse.html>

# Enumeración con nmap

## NSE (Nmap Scripting Engine)

- ▶ Podemos implementar nuestros propios script NSE.
- ▶ Si añadimos nuevos script a la carpeta */usr/share/nmap/scripts* o modificamos alguno de los existentes es necesario actualizar la base de datos de script (fichero ***script.db***).
  - `nmap --script-updatedb`

# Enumeración con nmap

## NSE (Nmap Scripting Engine)

- Un script NSE tiene una serie de campos:
- ***description***: una descripción para el usuario de lo que hace el script
  - ***categories***: define las categorías a las que pertenece el script.
  - ***author***: Los nombres de los autores e información de contacto.
  - ***license***: tipo de licencia atribuida al script.
  - ***dependencies***: un array con los scripts que deben ejecutarse antes que este script.
  - ***rules***: determina si el script debe ejecutarse contra el objetivo.
  - ***action***: la acción concreta que realiza el script.
  - ***environment variables***: suele emplearse para depuración.

<https://nmap.org/book/nse-script-format.html>

# Enumeración con nmap

## ¿Qué podemos enumerar?

- ▶ **Banner grabbing:** obtener tipo de software y versión tras un puerto abierto
- ▶ **Whois & traceroute** con geolocalización
- ▶ **Enumeración de ftp:** se permite login anónimo? Búsqueda de directorios comunes, búsqueda de backdoors, ...
- ▶ **Enumeración http:** métodos permitidos, directorios ocultos, comprobar existencia de WAF, ...
- ▶ **Enumeración SMB:** listas de usuarios, recursos, nombres de dominio, versiones, ...
- ▶ **Análisis de Vulnerabilidades:** smb-double-pulsar, smb-vulnms17-010 (eternal blue), ...

<https://www.infosecademy.com/nmap-smb-scripts-enumeration/>

# Enumeración con metasploit

## Metasploit

- ▶ Es un proyecto de código abierto.
- ▶ Proporciona información sobre vulnerabilidades y ayuda en test de penetración
- ▶ Herramienta compleja: en esta unidad nos centraremos sólo en la **enumeración**. En la siguiente, en **explotación**.
- ▶ Parte principal de metasploit → **módulos**.
- ▶ **Principales módulos**: exploits, **auxiliary**, payloads, encoders, nop generators, ...

# Enumeración con metasploit

## Metasploit

- ▶ Vamos a trabajar con módulos **auxiliary**
  - Lista completa: `search auxiliary`
- ▶ De entre estos, para enumeración nos interesan
  - **auxiliary/scanner**
- ▶ Ejemplos:
  - `auxiliary/scanner/enumusers`
  - `auxiliary/scanner/telnet/telnet_login`



# Enumeración con metasploit

## Usos de metasploit para escaneo y enumeración

- ▶ Escáneres de puertos:  
*auxiliary/scanner/portscan*
- ▶ Enumeración de servicios
  - Enumeración de SMB: versiones, usuarios, ...
  - Enumeración de HTTP
  - Enumeración de FTP
  - ...

# Análisis automático de vulnerabilidades

- ▶ Herramientas todo en 1:
  - Escaneo
  - Enumeración
  - Detección de vulnerabilidades
  - Generación de informes ...
- ▶ Facilitan la labor del auditor
- ▶ Existen tanto herramientas privativas como open source

# Análisis automático de vulnerabilidades

## OpenVAS

- ▶ Ahora se llama **GVM** (Greenbone Vulnerability Manager)
- ▶ Es gratuito y multiplataforma
- ▶ Actualmente no viene incluido en Kali
- ▶ Mantenido por Greenbone Networks, licencia GNU GPL
- ▶ **Características:**
  - Constantemente actualizado → NVT Feeds
  - Integración con OSSIM
  - Herramienta muy personalizable

# Análisis automático de vulnerabilidades

## Nessus

- ▶ También es multiplataforma
- ▶ NO es Open Source
- ▶ Permite una gran variedad de escáneres
- ▶ Es también un **compliance management** (capaz de chequear la política de seguridad de una organización)
- ▶ Características:
  - Rápido y potente
  - Permite añadir reglas y módulos personalizados
  - Es muy escalable
  - Escáner autenticado (perspectiva usuario interno) y no autenticado

# Análisis automático de vulnerabilidades

## Legion

- ▶ Gratuito y Open Source
- ▶ Muy sencillo
- ▶ Escrito en Python
- ▶ Incluido actualmente en Kali
- ▶ Se apoya en herramientas de terceros:
  - Nmap
  - Hydra
  - Nikto
  - Vulners
  - ...

# Escaneo y enumeración de redes y servicios

Fin