Práctica 5: Escaneo evasivo en redes corporativas simuladas

Objetivo

- Aprender técnicas de evasión (stealth scanning) para detectar servicios y hosts sin ser fácilmente detectado por firewalls o IDS.
- Simular un entorno empresarial con servicios básicos, reglas de firewall y, opcionalmente, un IDS.
- Analizar y comparar las diferentes técnicas, evaluando riesgos y posibles contramedidas.

1. Preparación del entorno

1.1. Máquina atacante (Kali Linux)

Herramientas: `nmap, masscan, wireshark, proxychains, netcat`

1.2. Máquina objetivo (*Ubuntu Server*)

- IP sugerida: `192.168.56.103`
- Servicios instalados:
- `sudo apt update && sudo apt install apache2 vsftpd samba openssh-server -y`
- Configuración del firewall (ufw):

```
sudo ufw enable
```

  - **Activa UFW**, que por defecto bloquea todas las conexiones entrantes y permite todas las salientes.
  - Si no estaba habilitado antes, empieza a filtrar tráfico inmediatamente.

```
sudo ufw allow 22
```

  - Abre el puerto **22/TCP**, usado por **SSH**.
  - Permite que puedas conectarte por SSH a la máquina (gestión remota).

```
sudo ufw allow 80
```

  - Abre el puerto **80/TCP**, usado por **HTTP (Apache2)**.
  - Permite que la máquina funcione como servidor web accesible.

```
sudo ufw allow 445
```

  - Abre el puerto **445/TCP**, usado por **SMB (**Samba en Linux, compartición de archivos en Windows).
  - Permite conexiones de red para compartir carpetas/recursos.

```
sudo ufw allow 21
```

- Abre el puerto **21/TCP**, usado por **FTP (vsftpd en este caso)**.
- Permite conexiones de clientes FTP hacia el servidor.

2. IDS en la práctica de escaneo evasivo

¿Cuál usar: Snort o Suricata?

Ambos son IDS muy usados, pero:

- **Snort** más clásico y ligero; tiene gran cantidad de reglas disponibles, ideal para aprender fundamentos.
- **Suricata** más moderno, soporta multihilo y análisis más avanzado; recomendado si quieres observar rendimiento en entornos más grandes.

Para esta **práctica académica y didáctica** utilizaremos **Snort**, porque:

- Tiene reglas simples de ejemplo.
- Es fácil ver cómo un escaneo Nmap dispara alertas.
- Hay más documentación en español para prácticas educativas.

Instalación del IDS

A) Snort

```
sudo apt install snort -y
```

Archivos importantes:

- Configuración: `/etc/snort/snort.conf`
- Reglas locales: `/etc/snort/rules/local.rules`
- Logs de alertas: `/var/log/snort/alert`

B) Suricata

```
sudo apt install suricata -y
```

Archivos importantes:

- Configuración: `/etc/suricata/suricata.yaml`
- Reglas: `/etc/suricata/rules/`
- Logs: `/var/log/suricata/fast.log`

3. Configuración básica de Snort

Edita el archivo de reglas locales:

```
sudo nano /etc/snort/rules/local.rules
```

Agrega reglas personalizadas:

A) Detectar escaneos SYN

```
alert tcp any any -> any 22 (flags:S; msg:"Posible escaneo SYN detectado en SSH";
sid:1000001;)
```

- Detecta paquetes TCP con flag **SYN** hacia el puerto 22.
- Mensaje de alerta: *Posible escaneo SYN*.

B) Detectar tráfico fragmentado

```
alert ip any any -> any any (fragoffset:>0; msg:"Paquete fragmentado detectado";
sid:1000002;)
```

- Detecta si llegan paquetes fragmentados (ej. `nmap -f`).
- Mensaje: *Paquete fragmentado detectado*.

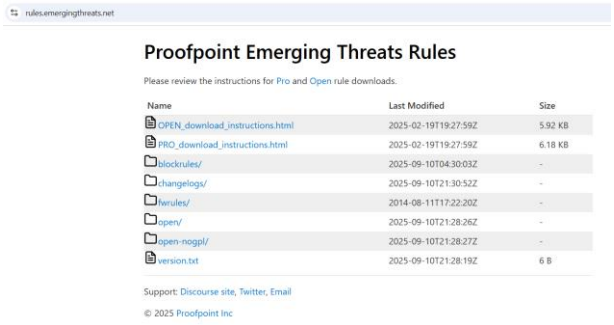
4. Configuración básica de Suricata

Para que Suricata detecte escaneos (Nmap, fragmentación, etc.), necesitas **descargar y habilitar un set de reglas**.

Instalar el paquete de reglas de Emerging Threats

```
sudo suricata-update
sudo suricata-update update-sources
sudo suricata-update
```

Estos comandos descargarán las reglas de Emerging Threats ¿Qué són?



Verificar la ruta de reglas en la config

Edita el archivo de configuración:

```
sudo nano /etc/suricata/suricata.yaml
```

Busca la sección, si no está añadela:

```
default-rule-path: /var/lib/suricata/rules
rule-files:
  - suricata.rules
- local.rules
```

Editar local.rules

```
sudo nano /var/lib/suricata/rules/local.rules
```

Y añade, por ejemplo:

```
# Detecta escaneo SYN al puerto SSH
alert tcp any any -> any 22 (flags:S; flow:to_server; msg:"[LOCAL] Escaneo SYN en SSH";
sid:1000001;)

# Detecta paquetes fragmentados
alert ip any any -> any any (fragoffset:>0; msg:"[LOCAL] Paquete fragmentado detectado";
sid:1000002;)

# Detecta escaneo al puerto FTP
alert tcp any any -> any 21 (flags:S; flow:to_server; msg:"[LOCAL] Escaneo FTP
detectado"; sid:1000003;)

# Detecta escaneo al puerto HTTP
alert tcp any any -> any 80 (flags:S; flow:to_server; msg:"[LOCAL] Escaneo HTTP
detectado"; sid:1000004;)

# Detecta escaneo al puerto SMB
alert tcp any any -> any 445 (flags:S; flow:to_server; msg:"[LOCAL] Escaneo SMB
detectado"; sid:1000005;)
```

Reiniciar Suricata con reglas

```
sudo systemctl restart suricata
```

4. Pruebas de funcionamiento

A) Ejecuta Snort en modo IDS:

```
sudo snort -A console -q -c /etc/snort/snort.conf -i enp0s3
```

(cambia `enp0s3` por tu interfaz de red)

- `-A console` muestra alertas en pantalla.
- `-q` modo silencioso (sin info extra).
- `-c` usa archivo de configuración.
- `-i` selecciona interfaz.

B) Ejecuta Suricata en modo IDS

Si prefieres probar Suricata, puedes activar su **modo en vivo**:

```
sudo suricata -c /etc/suricata/suricata.yaml -i enp0s3
```

Luego revisa los logs:

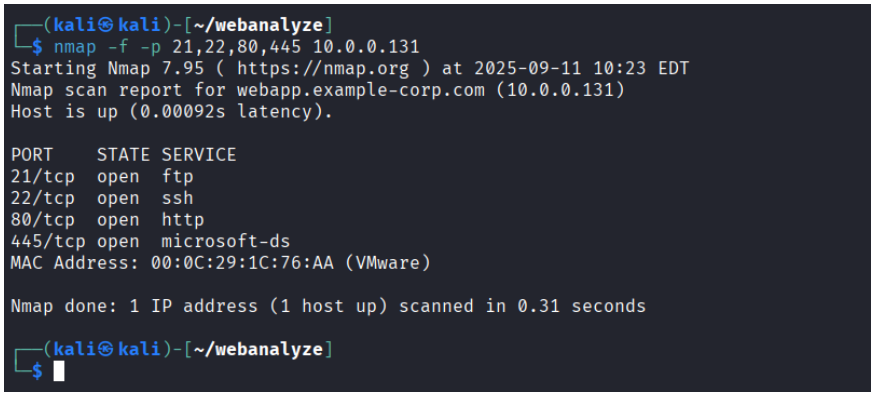
```
tail -f /var/log/suricata/fast.log
```

5. Ejercicios de escaneo evasivo con Nmap

A) Escaneo fragmentado

Envía paquetes fragmentados para dificultar la detección:

```
nmap -f -p 21,22,80,445 192.168.56.103
```



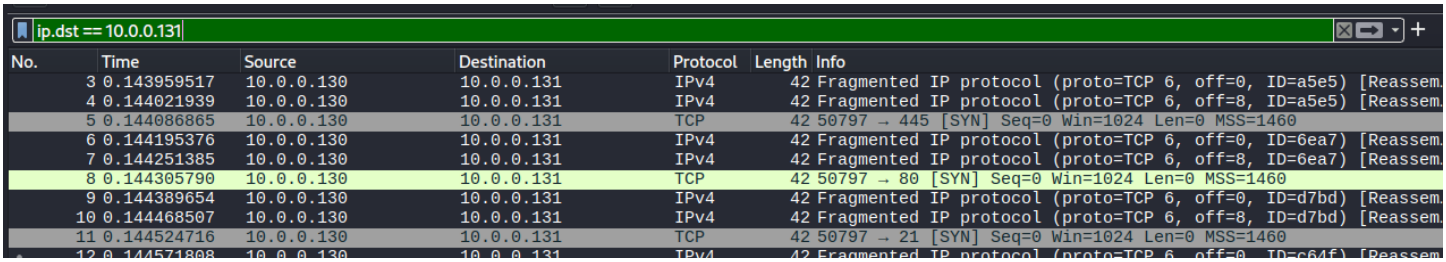
Comprobar si los puertos se detectan y si el IDS/firewall los registra.

```
ubuntu@ubuntu-server:/var/www/html/webapp$ sudo snort -A console -q -c /etc/snort/snort.conf -i ens33
09/11-14:23:58.621092  [**] [1:1000002:0] Paquete fragmentado detectado [**] [Priority: 0] {TCP} 10.0.0.130 -> 10.0.0.131
09/11-14:23:58.621366  [**] [1:1000001:0] Posible escaneo SYN detectado en SSH [**] [Priority: 0] {TCP} 10.0.0.130:57866 -> 10.0.0.131:22
09/11-14:23:58.621687  [**] [1:1000002:0] Paquete fragmentado detectado [**] [Priority: 0] {TCP} 10.0.0.130 -> 10.0.0.131
09/11-14:23:58.622409  [**] [1:1000002:0] Paquete fragmentado detectado [**] [Priority: 0] {TCP} 10.0.0.130 -> 10.0.0.131
09/11-14:23:58.623140  [**] [1:1000002:0] Paquete fragmentado detectado [**] [Priority: 0] {TCP} 10.0.0.130 -> 10.0.0.131
```

A**nalizar** con Wireshark y observar que los paquetes TCP aparecen divididos.

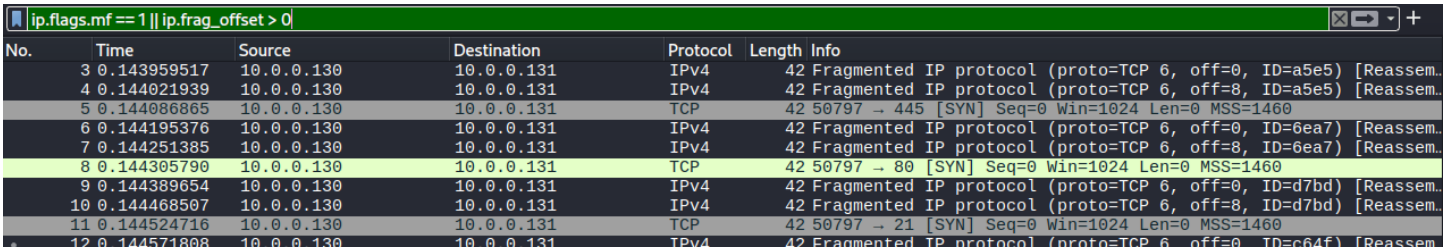
- Ver solo tráfico hacia la víctima:

```
ip.dst == 10.0.0.131
```



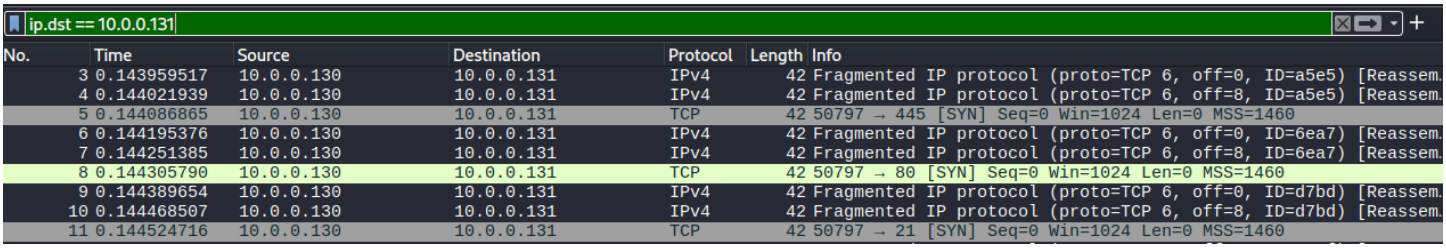
- Ver fragmentos del escaneo evasivo:

```
ip.flags.mf == 1 || ip.frag_offset > 0
```



- Ver intentos de conexión SSH (puerto 22):

```
tcp.port == 22
```



B) Genera un escaneo desde Kali

Escaneo SYN:

```
nmap -sS -p 22 192.168.56.103
```

envía paquetes SYN sin completar el handshake TCP (half-open scan). Comprobar si el puerto **22 (SSH)** está abierto y si el IDS/firewall lo registra. Para ello:

- En **Kali:** salida de Nmap (¿puerto abierto/cerrado?).
- En **Ubuntu (Snort):** alerta de "Posible escaneo SYN".
- En **Wireshark:** verás paquetes `SYN` sin `ACK` de confirmación.

B) Uso de Decoys (falsos atacantes)

Inyecta IPs señuelo para confundir al administrador de seguridad:

```
nmap -D RND:5 -p 21,22,80,445 192.168.56.103
```

mezcla tu IP real con 5 direcciones IP falsas (random) para confundir al administrador para que piense que hay múltiples atacantes. Compribamos:

- En **logs de Ubuntu:** conexiones de varias IPs distintas.
- En **Kali:** verás que Nmap sigue detectando servicios aunque "se esconda" entre decoys.
- En **Wireshark:** tráfico con IPs de origen falsas, interesante para comparar con tu IP real.

C) Timing evasivo (escaneo lento)

```
nmap -T1 -p- 192.168.56.103
```

escanea todos los puertos (`-p-`) pero con timing lento (`-T1`). Comparar un escaneo **sigiloso** (-T1) con uno **rápido** (-T4). Para ello comprobar:

- En **Kali:**
  - `-T4`: rápido, puede tardar segundos/minutos.
  - `-T1`: mucho más lento, pero más difícil de detectar.
- En **Snort:** reglas basadas en thresholds podrían no dispararse con `-T1`.
- En **Wireshark:** notarás menos "ráfagas" de paquetes; el tráfico se ve disperso en el tiempo.

D) Spoofing de MAC

Fingir otra tarjeta de red (ejemplo, Apple):

```
nmap --spoof-mac Apple -p 21,22,80 192.168.56.103
```

finge que la tarjeta de red es de otro fabricante (*en este caso Apple*). Verificar si en Wireshark aparece un fabricante distinto en la capa 2, para ello:

- En **Wireshark (Kali):** la dirección MAC de origen mostrará un OUI de Apple.
- En **Ubuntu:** puede no tener impacto en logs de firewall, porque estos suelen ver solo capa 3 (IP), no capa 2.
- Es más útil si simulas estar dentro de una LAN corporativa donde se monitorea el tráfico ARP/MAC.

6. Escaneo con Masscan

Instalación:

```
sudo apt install masscan -y
```

Ejemplo de escaneo:

```
masscan 192.168.56.0/24 -p21,22,80,445 --rate=100
```

comparar la velocidad y el nivel de detección con Nmap. Si usas un rate bajo, es más sigiloso; con un rate alto, IDS detectará tráfico anómalo.

7. Monitoreo con Wireshark

En Kali:

```
sudo wireshark
```

- Selecciona la interfaz de la red interna (ej: `eth1`).
- Filtro sugerido:

```
ip.addr == 192.168.56.103
```

- **Actividades de análisis:**
  - Identificar fragmentos de paquetes en el escaneo `-f`.
  - Comparar patrones de tráfico entre `nmap -T4` y `nmap -T1`.
  - Verificar la MAC spoofeada en los paquetes.

8. Actividades de evaluación

Actividad	Pasos a realizar	Preguntas guía
Escaneo fragmentado	Ejecuta nmap -f.	¿Se detectan los puertos? ¿El firewall bloqueó algo?
Escaneo SYN	Ejecuta nmap -sS.	¿detecta possible escaneo SYN?
Decoys	Ejecuta nmap -D.	¿Cuántos "atacantes" aparecen en los logs del servidor?
Timing	Compara -T1 y -T4.	¿Cuál fue más detectable? ¿Cuál más rápido?
Spoofing MAC	Ejecuta --spoof-mac.	¿Qué fabricante aparece en Wireshark? ¿Se engañó al sistema de registros?
Masscan vs Nmap	Escanea con ambos.	¿Cuál fue más rápido? ¿Cuál más sigiloso?
Informe de riesgos	Analiza resultados.	¿Qué cambios deberías aplicar en firewall/IDS para mejorar la defensa?

Informe final

1. **Introducción**
   - Objetivo de la práctica.
   - Descripción del entorno.
2. **Metodología**
   - Técnicas usadas (fragmentación, syn, decoys, timing, spoofing, masscan).
   - Configuración de firewall/IDS.
3. **Resultados**
   - Capturas de Nmap/Masscan.
   - Capturas de Wireshark.
   - Logs del firewall/IDS.
4. **Análisis comparativo**
   - Diferencias entre técnicas.
   - Ventajas y limitaciones.
5. **Recomendaciones defensivas**
   - Mejoras en firewall.
   - Reglas adicionales en IDS.
   - Estrategias de monitoreo.
6. **Conclusiones**
   - Lecciones aprendidas.
   - Reflexión sobre el balance entre ofensiva y defensiva.