



UNIVERSIDAD DEL ISTMO

CAMPUS TEHUANTEPEC

MATERIA:

Redes de Computadoras II.

DOCENTE:

I.C Carlos Mijangos Jiménez.

ALUMNO (A):

Karla Guadalupe Cordero Luna.

ACTIVIDAD: *MONITOREO Y ESCANEADO DE VULNERABILIDADES CON LAS HERRAMIENTAS NMAP Y WIRESHARK*

SEMESTRE:

7°

PARCIAL:

2°

CARRERA:

Ingeniería En Computación.

FECHA DE ENTREGA: LUNES 17 DE NOVIEMBRE DE 2025.

Índice

Introducción	3
Creación del Sandbox.....	3
Documentación de herramientas.....	5
<i>Nmap</i>	5
<i>Wireshark</i>	5
Desarrollo de la práctica.....	6
<i>Configuración previa</i>	6
<i>Escaneo de red con Nmap</i>	7
<i>Levantamiento de servicio para pruebas</i>	8
<i>Escaneo específico hacia el host detectado</i>	8
<i>Monitoreo del ataque en Wireshark</i>	9
Resultados esperados.....	11
Conclusiones	11
Referencias (APA).....	12

Introducción

La ciberseguridad es un área fundamental en la administración de redes. Para mantener sistemas seguros es necesario identificar vulnerabilidades y comportamientos anómalos que puedan representar amenazas.

En este proyecto se emplearon dos herramientas esenciales:

- **Nmap** para realizar reconocimiento de red y detección de servicios vulnerables
- **Wireshark** para monitorear tráfico y analizar paquetes en tiempo real

El objetivo principal es comprender cómo un atacante podría obtener información valiosa de una red y al mismo tiempo cómo un analista puede detectar estas acciones para implementar medidas de defensa.

Esta práctica se tiene otros objetivos secundarios:

- Detectar puertos y servicios activos en un entorno seguro (sandbox)
- Identificar métodos de escaneo más sigilosos
- Analizar tráfico generado para reconocer conexiones sospechosas
- Comprender el rol de Nmap en auditorías ofensivas y el de Wireshark en monitoreo defensivo

La combinación de estas herramientas permite realizar un ciclo ético de pentesting: **detección** → **explotación** → **monitoreo** → **mitigación**.

En seguridad informática es fundamental evaluar la exposición de servicios en red. Para ello, se emplean técnicas activas como escaneo de puertos con **Nmap**, complementadas con técnicas defensivas como inspección de tráfico mediante **Wireshark**.

Creación del Sandbox

Para realizar pruebas de forma segura se configuró un **entorno aislado** o *sandbox*:

Elemento	Función
Host: Windows con VirtualBox	Plataforma del laboratorio
VM atacante: ParrotOS	Ejecución de Nmap y Wireshark
VM víctima: Windows/Linux	Dispositivo objetivo

Configuración de red: Adaptador Puente o Host-Only	Permite comunicación directa entre las máquinas
--	---

- Host: Windows 10
- Sandbox: Máquina virtual **Parrot OS Security**
- Virtualización: VirtualBox
- Objetivo de ataque: el mismo equipo host en una red interna controlada

Comprobación de conectividad mediante ip addr y ipconfig para identificar direcciones internas:

- Host Windows: 192.168.0.21
- Parrot OS: 192.168.0.28

Este entorno garantiza que el análisis no afecte a la red real y permite el estudio de vulnerabilidades con control total.

```

[user@parrot]~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:d2:44:ae brd ff:ff:ff:ff:ff:ff
    inet6 2806:262:1428:232::2/128 scope global dynamic noprefixroute
        valid_lft 2586818sec preferred_lft 599618sec
    inet6 2806:262:1428:232:95b:de31:b3ec:1778/64 scope global dynamic noprefixroute
        valid_lft 2591984sec preferred_lft 604784sec
    inet6 fe80::d9e9:3ae6:f4dd:c489/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
[user@parrot]~$

```

```

Vínculo: dirección IPv6 local. . . . . : fe80::4abd:ceff:fe01:464c%2
Dirección IPv4. . . . . : 192.168.0.21
Máscara de subred . . . . . : 255.255.255.0
Puerta de enlace predeterminada . . . . . : fe80::4abd:ceff:fe01:464c%2
192.168.0.1

```

Documentación de herramientas

Nmap

Nmap (**Network Mapper**) es un escáner de redes utilizado para:

- Descubrir hosts conectados
- Identificar servicios y puertos abiertos
- Detectar sistemas operativos
- Encontrar vulnerabilidades conocidas

Funciones utilizadas

Técnica	Parámetro	Propósito
Escaneo completo de puertos	-p-	Detectar todos los puertos TCP
Escaneo sigiloso	-sS	Evitar 3-way handshake completo
Aumento de velocidad	--min-rate 5000	Paquetes por segundo
Nivel de detalle	-vvv	Verbose máximo
Sin DNS	-n	Mayor rapidez
Forzar sin sping	-Pn	Si el host no responde ICMP

Wireshark

Wireshark es un analizador de tráfico que permite:

- Capturar paquetes en tiempo real
- Identificar protocolos y sesiones activas
- Detectar escaneos y comportamiento sospechoso

Características vistas:

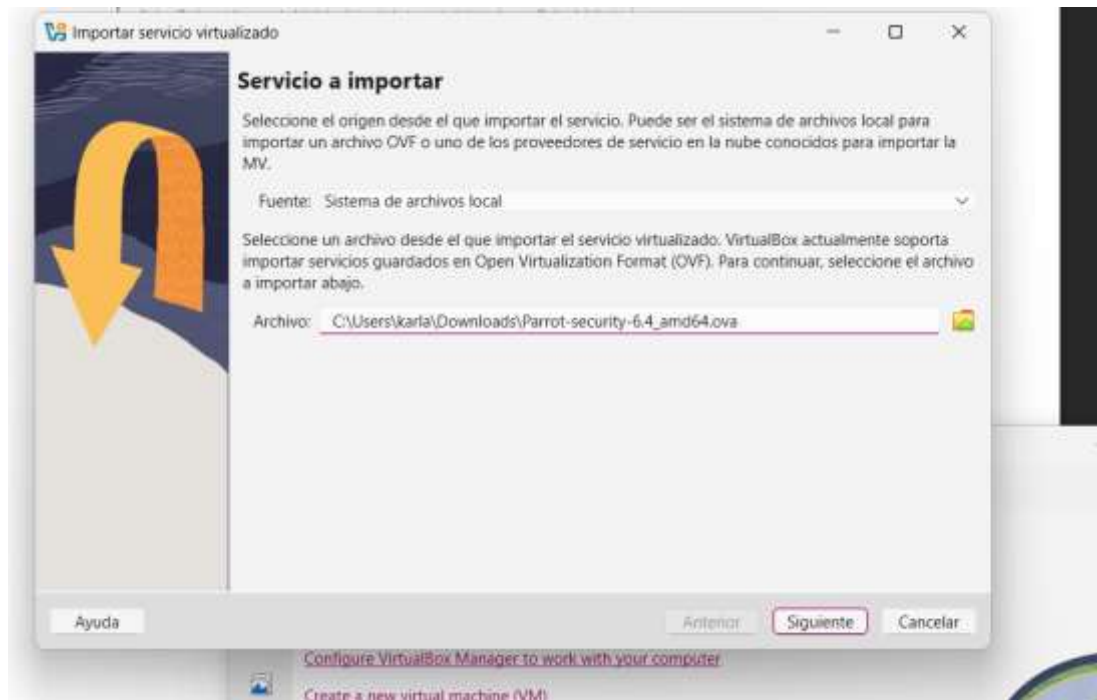
- Filtro por dirección IP:
ip.addr == 192.168.0.21
- Filtro por puerto específico:
tcp.port == 80
- Identificación del handshake TCP y respuesta HTTP

Wireshark permite al analista reconocer patrones de ataque o actividad no autorizada en la red.

Desarrollo de la práctica

Configuración previa

1. Se importó ParrotOS en VirtualBox (.ova)



2. Se configuró red en modo **Adaptador Puente**



3. Se probó la conexión

4. Se inició captura de tráfico en Wireshark antes del escaneo

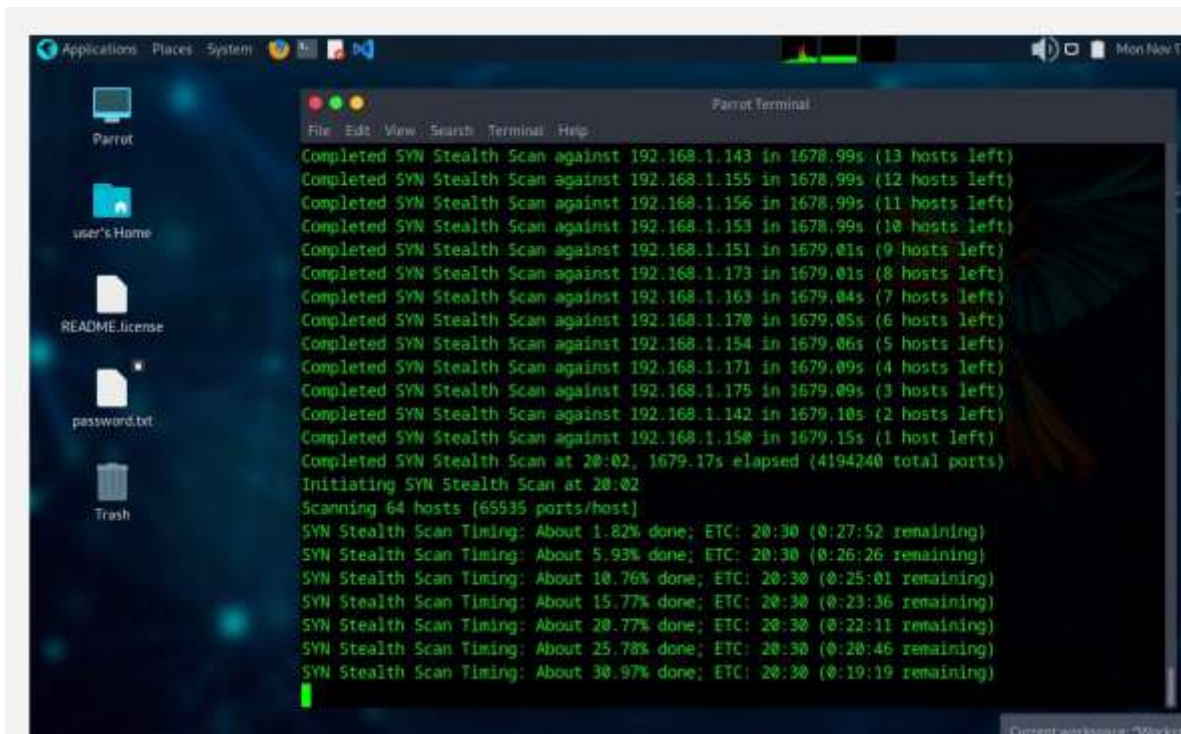
Escaneo de red con Nmap

Se utilizó el siguiente escaneo agresivo/rápido:

```
nmap -p- --open -sS --min-rate 5000 -vvv -n -Pn 192.168.1.0/24
```

- Se identificó al host **192.168.0.21** como activo con el **puerto 80 abierto**.
- Posible servicio HTTP en ejecución.

Resultado del escaneo de red (hosts detectados)

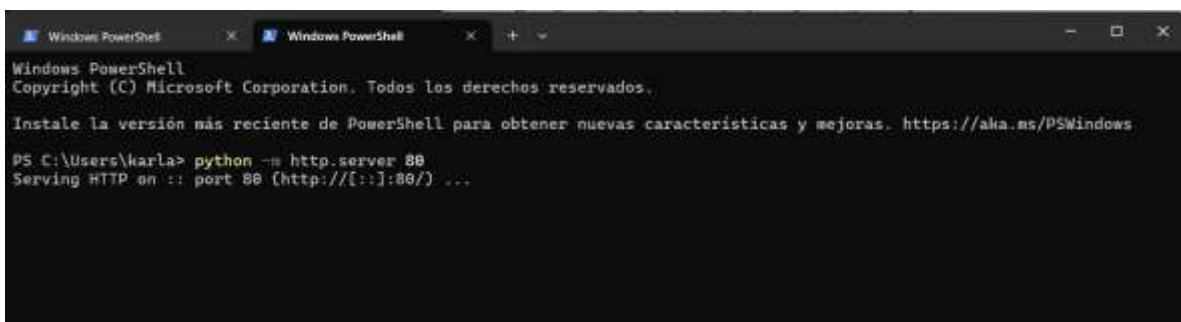


Levantamiento de servicio para pruebas

En Windows se activó un servidor web local:

`python -m http.server 80`

Confirmando entonces que el servicio en el **puerto 80 era HTTP**.



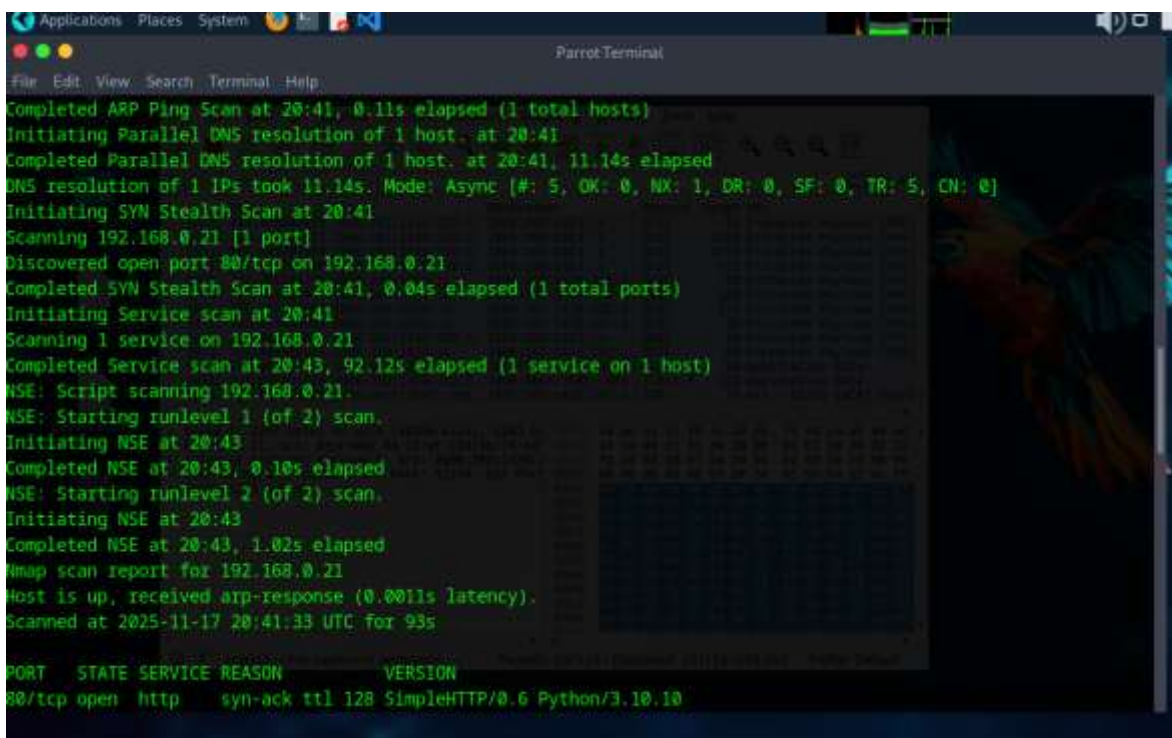
Escaneo específico hacia el host detectado

`sudo nmap -p 80 -sS -sV -vvv 192.168.0.21`

Puerto 80 abierto, sirviendo:

SimpleHTTP/0.6 Python 3.10.10

Resultado del escaneo al puerto 80:



```
Completed ARP Ping Scan at 20:41, 0.11s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 20:41
Completed Parallel DNS resolution of 1 host. at 20:41, 11.14s elapsed
DNS resolution of 1 IPs took 11.14s. Mode: Async [#: 5, OK: 0, NX: 1, DR: 0, SF: 0, TR: 5, CN: 0]
Initiating SYN Stealth Scan at 20:41
Scanning 192.168.0.21 [1 port]
Discovered open port 80/tcp on 192.168.0.21
Completed SYN Stealth Scan at 20:41, 0.04s elapsed (1 total ports)
Initiating Service scan at 20:41
Scanning 1 service on 192.168.0.21
Completed Service scan at 20:43, 92.12s elapsed (1 service on 1 host)
NSE: Script scanning 192.168.0.21.
NSE: Starting runlevel 1 (of 2) scan.
Initiating NSE at 20:43
Completed NSE at 20:43, 0.10s elapsed
NSE: Starting runlevel 2 (of 2) scan.
Initiating NSE at 20:43
Completed NSE at 20:43, 1.02s elapsed
Nmap scan report for 192.168.0.21
Host is up, received arp-response (0.0011s latency).
Scanned at 2025-11-17 20:41:33 UTC for 93s

PORT      STATE SERVICE REASON          VERSION
80/tcp    open  http    syn-ack ttl 128 SimpleHTTP/0.6 Python/3.10.10
```

Esto valida que el escaneo de red fue preciso.

Monitoreo del ataque en Wireshark

Se aplicó el filtro:

ip.addr == 192.168.0.21 && tcp.port == 80

Paquetes observados:

1. **SYN → SYN/ACK → ACK**
Establecimiento de conexión TCP
2. **GET /**
Solicitud HTTP
3. **HTTP/1.1 200 OK**
Respuesta del servidor con texto

Capturing from enp0s3 (as superuser)

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

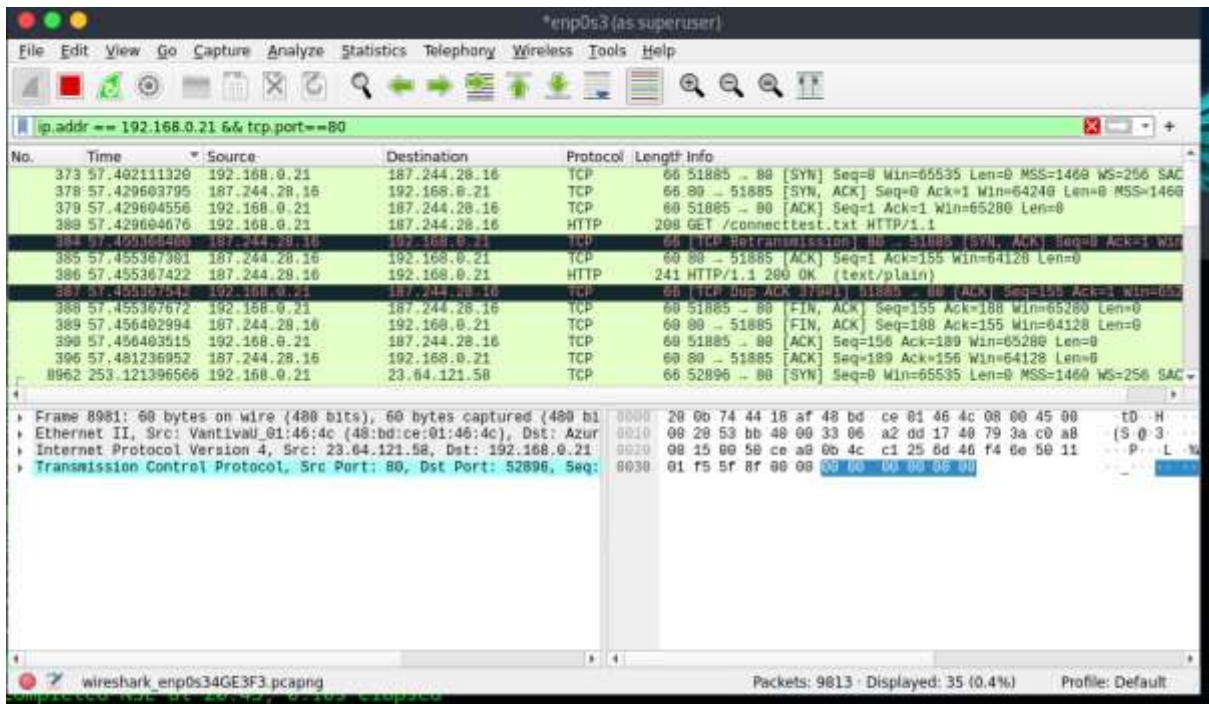
No.	Time	Source	Destination	Protocol	Length	Info
1135...	764.414271738	2806:260:1023::c	2806:262:1420:232:3...	UDP	88	443 → 62838 Len=26
1135...	764.989564571	2806:260:1023::c	2806:262:1420:232:3...	UDP	109	443 → 62838 Len=47
1135...	764.992136764	2806:262:1420:232:3...	2806:260:1023::c	UDP	95	62838 → 443 Len=33
1135...	766.732413156	140.82.113.5	192.168.0.21	TLSv1.3	93	Application Data
1135...	766.732413837	192.168.0.21	140.82.113.5	TCP	60	53812 → 443 [FIN, ACK]
1135...	766.817007500	140.82.113.5	192.168.0.21	TLSv1.3	78	Application Data
1135...	766.817008362	140.82.113.5	192.168.0.21	TCP	60	443 → 53812 [FIN, ACK]
1135...	766.817008492	192.168.0.21	140.82.113.5	TCP	60	53812 → 443 [RST, ACK]
1135...	766.825438644	2a03:2880:f235:1cd:...	2806:262:1420:232:3...	TLSv1.3	599	Application Data
1135...	766.866770257	2806:262:1420:232:3...	2a03:2880:f235:1cd:...	TCP	74	57702 → 443 [ACK] Seq=6
1135...	766.938456154	2806:262:1420:232:3...	2a03:2880:f235:1cd:...	TLSv1.3	172	Application Data
1135...	766.986294140	2a03:2880:f235:1cd:...	2806:262:1420:232:3...	TCP	74	443 → 57702 [ACK] Seq=3
1135...	767.017611290	2806:262:1420:232:3...	2806:260:1023::c	UDP	1287	62838 → 443 Len=1225

Frame 1: 1287 bytes on wire (10296 bits), 1287 bytes captured (10296 bits) on interface enp0s3
Ethernet II, Src: AzureWav_44:18:af (20:0b:74:44:18:af), Dst: 08:00:27:00:00:00
Internet Protocol Version 6, Src: 2806:262:1420:232:3::3, Dst: 2806:260:1023::c
User Datagram Protocol, Src Port: 57174, Dst Port: 443
Data (1225 bytes)

0000 48 bd ce 01 46 4c 20 0b 74 44 18 af 86 dd
0010 a3 6a 04 d1 11 40 28 06 02 62 14 20 02 32
0020 fb d4 95 59 87 8a 28 06 02 60 10 23 00 00
0030 00 00 00 00 00 0f df 56 01 bb 04 d1 ec f0
0040 7e 73 ba 2e 50 c1 e5 3b a8 4d 11 e6 15 bf
0050 41 24 0f e3 cb 60 b2 fb ba 46 1b 8c 71 51
0060 96 e3 7a 47 58 f3 bc 3e 6e a0 58 31 9a 40
0070 77 91 9c 62 28 f8 00 98 22 c9 21 44 6a 48
0080 ac 95 7c 88 3b 3d bc 3b a5 3c bc 40 9d 39
0090 84 3a fb be 8b f9 94 c5 7b 08 2a 0f fe c0
00a0 33 e8 82 0e 86 49 46 1e 6d 61 04 b9 f7 bc
00b0 62 ae 26 31 e6 f5 01 13 89 03 e0 42 60 7d
00c0 ed df 16 78 07 00 04 2c 99 48 de 7b f7 10
00d0 de 6d 62 c2 51 54 10 24 00 df 6a 5b ae de
00e0 08 08 12 4b 00 00 c2 81 6d 26 c0 5f 02 7a

enp0s3: <live capture in progress> Packets: 115546 · Displayed: 115546 (100.0%) Profile: Default

Completed NSE at 20:43, 0.165 elapsed
NSE: Starting runlevel 2 (of 2) scan.
Initiating NSE at 20:43



Resultados esperados

- Se identificaron hosts activos en la red
- Se detectaron puertos abiertos en la máquina víctima
- Se reconocieron servicios asociados

Esto demuestra que un administrador puede detectar escaneos si monitorea activamente la red.

Conclusiones

El uso combinado de Nmap y Wireshark permitió identificar y analizar servicios activos dentro de un entorno seguro, comprobando en la práctica la importancia del escaneo de puertos en auditorías de seguridad, así como del monitoreo de red para la detección de actividad anómala.

Nmap facilitó la enumeración de hosts y la identificación de un servicio HTTP en el puerto 80 del equipo objetivo mediante un escaneo sigiloso y completo, mientras que Wireshark confirmó la comunicación mediante la captura del proceso de establecimiento de conexión TCP y solicitudes HTTP reales, demostrando que la información recolectada es confiable y verificable.

Gracias a esta metodología ofensiva-defensiva, se reconoce la eficacia de ambas herramientas en la evaluación y protección de redes, reforzando la importancia del monitoreo constante para prevenir vulnerabilidades y posibles ataques.

Referencias

García, D. (2023). *Auditoría de redes con Nmap*. Editorial Alfaomega.

Nmap. (2025). *Nmap Reference Guide*. <https://nmap.org/book/man.html>

Wireshark. (2025). *User Documentation*. <https://www.wireshark.org/docs/>

Rouse, M. (2024). *What is a sandbox environment?*. TechTarget.