

Tecnologia de Segurança (4º ano de Engenharia Informática)

Trabalho Prático Nº2 Parte B

Grupo 7

Relatório de Desenvolvimento

Carla Cruz
(a80564)

Adriana Meireles
(a82582)

6 de Dezembro de 2019

Conteúdo

1	Introdução	2
2	Parte 1	3
2.0.1	Q1	3
2.0.2	Q2	9
2.0.3	Q3	18
2.0.4	Q4	19
3	Parte 2	23
3.0.1	Q5	23
3.0.2	Q6	25
3.0.3	Q7	28
3.0.4	Q8	29
4	Conclusão	32

Capítulo 1

Introdução

O trabalho anterior consistia na investigação de empresas através de scanning passivo. Neste enunciado, utilizamos técnicas de scanning ativo ou *PenTesting*. Para isso, foram utilizadas as ferramentas WireShark, Snort, Nessus e Nmap.

Com a ajuda destes recursos, o principal objetivo seria identificar vulnerabilidades e, consequentemente, apresentar soluções para a atenuação das mesmas.

Capítulo 2

Parte 1

2.0.1 Q1

O objetivo desta questão é utilizar o nmap para fazer diferentes tipos de scans a um determinado alvo. Neste caso o alvo foi a máquina de testes *metaesploitable 2*. Para além da análise dos resultados gerados pelo nmap também se observa o tráfego gerado com a ferramenta Wireshark.

1) nmap -sn target

Este comando é responsável pela não realização de um port scan depois de descobrir um host. É normalmente usado no reconhecimento de uma target network rapidamente sem dar muito nas vistas. Resumidamente esta flag é responsável por efetuar um ping ao nosso target para ver apenas se este se encontra "vivo".

```
root@kali:~# nmap -sn 172.16.7.2
Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-29 22:54 WET
Profile: Default
Nmap scan report for 172.16.7.2
Host is up (0.00046s latency).
MAC Address: 08:00:27:ED:DA:41 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 0.05 seconds
```

Figura 2.1: Resultado do comando nmap -sn

O scan foi muito rápido(0.05 segundos) e para além de nos indicar que o alvo estava online também nos indicou o Mac Address.

Pode observar-se o tráfego no wireshark após o comando nmap -sn 172.16.7.2:

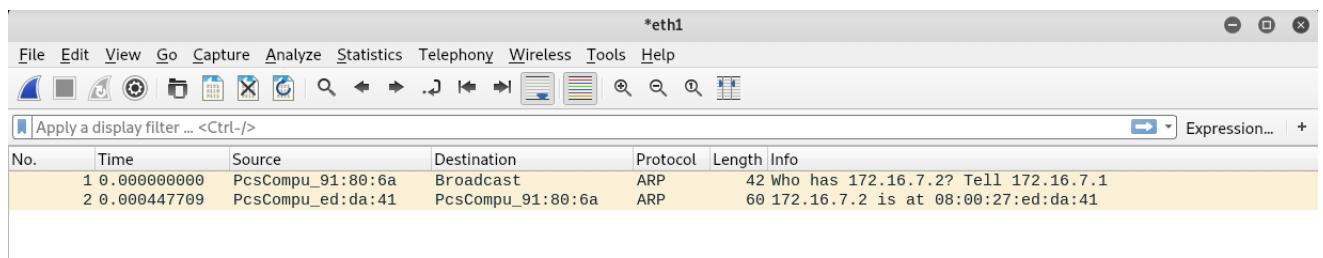


Figura 2.2: Resultado do comando nmap -sn no wireshark

2) nmap -sU target

É utilizada a flag -sU. Esta flag indica ao nmap para fazer um scan UDP. Este tipo de scan pode ser combinado com um TCP scan como o SYN scan e verificar simultaneamente ambos os protocolos. Este scan envia um pacote UDP para cada port alvo. Caso recebamos um pacote de ICMP do tipo 3, código 3, significa que aquela porta está indisponível. Se recebermos um pacote do tipo 3, código 0,1,2,9,10 ou 13 indica que as portas estão filtered. Se não obtivermos resposta poderá indicar que a porta está aberta mas o tráfego está a ser filtrado. Por último, se recebermos um pacote UDP é porque aquela porta está aberta.

```
root@kali:~# nmap -sU 172.16.7.2
Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-29 22:57 WET
Nmap scan report for 172.16.7.2
Host is up (0.00079s latency).

Not shown: 993 closed ports
PORT      STATE SERVICE
53/udp    open  domain
68/udp    open|filtered dhcpc
69/udp    open|filtered tftp
111/udp   open  rpcbind
137/udp   open  netbios-ns
138/udp   open|filtered netbios-dgm
2049/udp  open  nfs
MAC Address: 08:00:27:ED:DA:41 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 1083.47 seconds
```

Figura 2.3: Resultado do comando nmap -sU

Foram encontrados inúmeros serviços neste tipo de scan, que embora seja lento, é crucial na medida em que existem muitas vulnerabilidades em serviços UDP. Dura muito tempo devido à limitação da taxa de respostas ICMP do Linux.

Pode observar-se o tráfego no wireshark após o comando nmap -sU 172.16.7.2:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	PcsCompu_91:80:6a	Broadcast	ARP	42	Who has 172.16.7.2? Tell 172.16.7.1
2	0.000393930	PcsCompu_ed:da:41	PcsCompu_91:80:6a	ARP	60	172.16.7.2 is at 08:00:27:ed:da:41
3	0.026969003	172.16.7.1	172.16.7.2	UDP	42	46894 → 2222 Len=0
4	0.027024648	172.16.7.1	172.16.7.2	UDP	42	46894 → 19650 Len=0
5	0.027047408	172.16.7.1	172.16.7.2	UDP	42	46894 → 21524 Len=0
6	0.027068165	172.16.7.1	172.16.7.2	UDP	42	46894 → 17683 Len=0
7	0.027099726	172.16.7.1	172.16.7.2	UDP	42	46894 → 39683 Len=0
8	0.027111857	172.16.7.1	172.16.7.2	UDP	42	46894 → 18582 Len=0
9	0.027133980	172.16.7.1	172.16.7.2	UDP	42	46894 → 21655 Len=0
10	0.027159434	172.16.7.1	172.16.7.2	UDP	42	46894 → 44923 Len=0
11	0.027180850	172.16.7.1	172.16.7.2	NFS	82	V2 NULL Call (Reply In 19)
12	0.027202037	172.16.7.1	172.16.7.2	UDP	42	46894 → 20522 Len=0
13	0.027391700	172.16.7.2	172.16.7.1	ICMP	70	Destination unreachable (Port unreachable)
14	0.027400725	172.16.7.2	172.16.7.1	ICMP	70	Destination unreachable (Port unreachable)
15	0.027402107	172.16.7.2	172.16.7.1	ICMP	70	Destination unreachable (Port unreachable)
16	0.027403500	172.16.7.2	172.16.7.1	ICMP	70	Destination unreachable (Port unreachable)
17	0.027404638	172.16.7.2	172.16.7.1	ICMP	70	Destination unreachable (Port unreachable)
18	0.027405791	172.16.7.2	172.16.7.1	ICMP	70	Destination unreachable (Port unreachable)
19	0.027576809	172.16.7.2	172.16.7.1	NFS	66	V2 NULL Reply (Call In 11)
20	0.027593986	172.16.7.1	172.16.7.2	ICMP	94	Destination unreachable (Port unreachable)
21	0.029884209	172.16.7.1	172.16.7.2	UDP	42	46894 → 21649 Len=0
22	0.029927884	172.16.7.1	172.16.7.2	UDP	42	46894 → 53037 Len=0
23	0.029952418	172.16.7.1	172.16.7.2	UDP	42	46894 → 21320 Len=0
24	0.029973563	172.16.7.1	172.16.7.2	UDP	42	46894 → 687 Len=0

Figura 2.4: Resultado do comando nmap -sU no wireshark

Como explicado anteriormente, foram enviados muitos pacotes UDP para a target do metasploitable e

conseguimos ver que a porta que queremos está fechada devido às respostas de pacotes ICMP do tipo *Unreachable*.

udp.port == 111								
No.	Time	Source	Destination	Protocol	Length	Source Port	Destination Port	
760	284.395977661	172.16.7.2	172.16.7.1	Portmap	74	111	46894	
761	284.396025145	172.16.7.1	172.16.7.2	ICMP	102	111	46894	
759	284.395031971	172.16.7.1	172.16.7.2	Portmap	82	46894	111	

Figura 2.5: Resultado do comando nmap -sU no wireshark

Observando os dados anteriores, encontrámos um caso em que a porta que queremos está aberta, respondendo com um pacote do tipo Portmap.

3) nmap -sT target

É utilizada a flag -sT. Esta flag deve ser utilizada quando o *TCP SYN* scan não é possível (por exemplo, quando se está a analisar uma rede *ipv6*). Através da system call *connect*, o nmap pede ao sistema operativo para estabelecer a conexão com a target. É usada, pelo nmap, a interface *Berkeley Sockets API* para se saber o estado em cada tentativa de conexão. É um tipo de scan pouco eficiente pois necessita de completar uma conexão para abrir as portas da target e, portanto, necessita de mais pacotes para obter a mesma informação o que pode levar as targets à colocação das conexões nos logs. Caso essas conexões tenham vindo de um único sistema permite ao administrador saber se foi scanned. É, deste modo, um scan mais ruidoso e detetável.

```

root@kali:~# nmap -sT 172.16.7.2
Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-29 23:40 WET
Nmap scan report for 172.16.7.2
Host is up (0.00051s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 08:00:27:ED:DA:41 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.15 seconds

```

Figura 2.6: Resultado do comando nmap -sT

Através da captura anterior conseguimos observar quais serviços estavam a correr nas respetivas portas abertas.

Pode observar-se o tráfego no wireshark após o comando nmap -sT 172.16.7.2:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.0000000000	PcsCompu_91:80:6a	Broadcast	ARP	42	Who has 172.16.7.2? Tell 172.16.7.1
2	0.000291664	PcsCompu_ed:da:41	PcsCompu_91:80:6a	ARP	60	172.16.7.2 is at 08:00:27:ed:da:41
3	0.016875421	172.16.7.1	172.16.7.2	TCP	74	53272 → 199 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval...
4	0.016934611	172.16.7.1	172.16.7.2	TCP	74	49186 → 1723 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSva...
5	0.016966625	172.16.7.1	172.16.7.2	TCP	74	51594 → 110 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval...
6	0.016996942	172.16.7.1	172.16.7.2	TCP	74	35298 → 1025 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSva...
7	0.017077670	172.16.7.1	172.16.7.2	TCP	74	49832 → 993 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval...
8	0.017100102	172.16.7.2	172.16.7.1	TCP	60	199 → 53272 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
9	0.017106509	172.16.7.2	172.16.7.1	TCP	60	1723 → 49186 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
10	0.017152263	172.16.7.1	172.16.7.2	TCP	74	50710 → 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval...
11	0.017196154	172.16.7.2	172.16.7.1	TCP	60	110 → 51594 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
12	0.017199679	172.16.7.2	172.16.7.1	TCP	60	1025 → 35298 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
13	0.017200846	172.16.7.2	172.16.7.1	TCP	60	993 → 49832 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
14	0.017247889	172.16.7.1	172.16.7.2	TCP	74	34952 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval...
15	0.017346513	172.16.7.1	172.16.7.2	TCP	74	37240 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval...
16	0.017401471	172.16.7.1	172.16.7.2	TCP	74	53322 → 587 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval...
17	0.017477551	172.16.7.1	172.16.7.2	TCP	74	37210 → 111 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval...
18	0.017587437	172.16.7.2	172.16.7.1	TCP	74	22 → 56710 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PER...
19	0.017598096	172.16.7.1	172.16.7.2	TCP	66	50710 → 22 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1928257261 TSec...
20	0.017607004	172.16.7.2	172.16.7.1	TCP	74	80 → 34952 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PER...
21	0.017610095	172.16.7.1	172.16.7.2	TCP	66	34952 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1928257261 TSec...
22	0.017679629	172.16.7.2	172.16.7.1	TCP	74	445 → 37240 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK PE...
23	0.017684486	172.16.7.1	172.16.7.2	TCP	66	37240 → 445 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1928257261 TSe...
24	0.017691812	172.16.7.2	172.16.7.1	TCP	60	587 → 53322 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
25	0.017744168	172.16.7.1	172.16.7.2	TCP	66	50710 → 22 [RST, ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1928257261...
26	0.017872668	172.16.7.2	172.16.7.1	TCP	74	111 → 37210 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK PE...
27	0.017882521	172.16.7.1	172.16.7.2	TCP	66	37210 → 111 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1928257261 TSe...

Figura 2.7: Resultado do comando nmap -sT no wireshark

Na captura anterior, é possível observar a dinâmica de uma conexão TCP completa quando a porta alvo está aberta. Caso a porta alvo esteja fechada, é enviado um pacote RST a partir do target.

4) nmap -sS target

O tipo de scan -sS é conhecido como *TCP SYN*, é o default do nmap e mais popular por vários motivos. Pode ser realizado rapidamente e é capaz de analisar milhares de porta por segundo. Para além do referido anteriormente, é o scan mais silencioso pois nunca completa as conexões TCP. Caso recebamos SYN-ACK então a porta está aberta. Caso recebamos um TCP RST significa que a porta está fechada. Por fim, se recebermos um pacote ICMP do tipo unreachable, ou não recebermos nada significa que a porta está filtrada.

```

root@kali:~# nmap -sS 172.16.7.2
Starting Nmap 7.80 (https://nmap.org) at 2019-11-29 23:42 WET
Nmap scan report for 172.16.7.2
Host is up (0.00016s latency).:80:6a
Not shown: 5973 closed ports
MAC Address: 08:00:27:ED:DA:41 (Oracle VirtualBox virtual NIC)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown

Nmap done: 1 IP address (1 host up) scanned in 0.16 seconds

```

Figura 2.8: Resultado do comando nmap -sS

Como podemos observar o resultado é semelhante ao da flag -sT pois, apesar de ambas executarem de maneiras diferentes, possuem a mesma finalidade.

Pode observar-se o tráfego no wireshark após o comando nmap -sS 172.16.7.2:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0xfb01ee23
2	0.314325062	PcsCompu_91:80:6a	Broadcast	ARP	42	Who has 172.16.7.2? Tell 172.16.7.1
3	0.314695686	PcsCompu_ed:da:41	PcsCompu_91:80:6a	ARP	60	172.16.7.2 is at 08:00:27:ed:da:41
4	0.331992848	172.16.7.1	172.16.7.2	TCP	58	46375 → 587 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
5	0.332051514	172.16.7.1	172.16.7.2	TCP	58	46375 → 3389 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
6	0.332078877	172.16.7.1	172.16.7.2	TCP	58	46375 → 554 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
7	0.332102181	172.16.7.1	172.16.7.2	TCP	58	46375 → 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
8	0.332125530	172.16.7.1	172.16.7.2	TCP	58	46375 → 8888 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
9	0.332148530	172.16.7.1	172.16.7.2	TCP	58	46375 → 25 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
10	0.332175692	172.16.7.1	172.16.7.2	TCP	58	46375 → 21 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11	0.332200372	172.16.7.1	172.16.7.2	TCP	58	46375 → 1025 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
12	0.332223991	172.16.7.1	172.16.7.2	TCP	58	46375 → 111 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
13	0.332247808	172.16.7.1	172.16.7.2	TCP	58	46375 → 143 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
14	0.332273533	172.16.7.2	172.16.7.1	TCP	60	587 → 46375 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
15	0.332377462	172.16.7.2	172.16.7.1	TCP	60	3389 → 46375 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
16	0.332381622	172.16.7.2	172.16.7.1	TCP	60	554 → 46375 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
17	0.332383978	172.16.7.2	172.16.7.1	TCP	60	53 → 46375 [SYN, ACK] Seq=0 Ack=0 Win=5840 Len=0 MSS=1460
18	0.332391026	172.16.7.2	172.16.7.1	TCP	54	46375 → 53 [RST] Seq=1 Win=0 Len=0
19	0.332403452	172.16.7.2	172.16.7.1	TCP	60	8888 → 46375 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
20	0.332406935	172.16.7.2	172.16.7.1	TCP	60	25 → 46375 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
21	0.332410820	172.16.7.1	172.16.7.2	TCP	54	46375 → 25 [RST] Seq=1 Win=0 Len=0
22	0.332564623	172.16.7.2	172.16.7.1	TCP	60	21 → 46375 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
23	0.332572689	172.16.7.1	172.16.7.2	TCP	54	46375 → 21 [RST] Seq=1 Win=0 Len=0
24	0.332582507	172.16.7.2	172.16.7.1	TCP	60	1025 → 46375 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
25	0.332585570	172.16.7.2	172.16.7.1	TCP	60	111 → 46375 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
26	0.332589104	172.16.7.1	172.16.7.2	TCP	54	46375 → 111 [RST] Seq=1 Win=0 Len=0
27	0.332595582	172.16.7.2	172.16.7.1	TCP	60	143 → 46375 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
28	0.332653398	172.16.7.1	172.16.7.2	TCP	58	46375 → 5900 [SYN] Seq=0 Win=1024 Len=0 MSS=1460

Figura 2.9: Resultado do comando nmap -sS no wireshark

Observando a partir da linha 14, sempre que é recebida uma resposta do tipo SYN/ACK, é enviada uma resposta do tipo RST que indica o desfecho da ligação. Desta forma, como o handshake não é completo, foi obtido um menor volume de pacotes, utilizando-se menos um pacote por ligação que nas flags anteriores. Assim, concluímos que é um scan mais discreto.

2.0.2 Q2

Esta questão tinha o mesmo objetivo da anterior, com a diferença que nesta teríamos de fazer um scan tanto à máquina vulnerável metasploitable 2 como ao servidor com endereço IP 45.33.32.156. Desta forma, através da utilização do comando ping prova-se que a conexão existe:

```
root@kali:~# ping 45.33.32.156      172.16.7.1      TCP
PING 45.33.32.156 (45.33.32.156) 56(84) bytes of data.
64 bytes from 45.33.32.156: icmp_seq=1 ttl=63 time=191 ms
64 bytes from 45.33.32.156: icmp_seq=2 ttl=63 time=171 ms
64 bytes from 45.33.32.156: icmp_seq=3 ttl=63 time=179 ms
^C27 0.332595582  172.16.7.2      172.16.7.1      TCP
--245.33.32.156 ping statistics --  172.16.7.2      TCP
3 packets transmitted, 3 received, 0% packet loss, ptime 2003ms
rtt min/avg/max/mdev = 171.188/180.253/190.930/8.139 ms
```

Figura 2.10: Resultado do comando ping ao servidor

1) nmap -sA target

Este tipo de scan não determina as portas abertas. Tem várias aplicações tais como: descobrir conjuntos de regras das firewalls, determinar se tem estados e quais as portas filtradas. O pacote de probe tem apenas a ACK flag ligada, daí este tipo de scan chamar-se *TCP ACK* scan. As ports abertas e fechadas vão devolver um pacote RST caso se faça um scan a sistemas não filtrados. Posteriormente, é colocada pelo nmap uma label nessas ports como unfiltered significando que o pacote ACK não sabe se a port está aberta ou fechada.

```

root@kali:~# nmap -sA 172.16.7.2           172.16.7.2      TCP
Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-29 23:49 WET
Nmap scan report for 172.16.7.2           172.16.7.2      TCP
Host is up (0.00019s latency).           172.16.7.2      TCP
All 1000 scanned ports on 172.16.7.2 are unfiltered      TCP
MAC Address: 08:00:27:ED:DA:41 (Oracle VirtualBox virtual NIC)CP
Name 26: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) o
Name 26: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) o
Nmap done: 1 IP address (1 host up) scanned in 0.15 seconds

```

Figura 2.11: Resultado do comando nmap -sA ao metasploitable

```

root@kali:~# nmap -sA 45.33.32.156          10.0.2.15      TCP
Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-29 23:53 WET
Nmap scan report for scanme.nmap.org (45.33.32.156)          10.0.2.15      TCP
Host is up (0.000041s latency).           10.0.2.15      TCP
All 1000 scanned ports on scanme.nmap.org (45.33.32.156) are unfiltered      TCP
Name 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface
Name 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface
Nmap done: 1 IP address (1 host up) scanned in 0.17 seconds

```

Figura 2.12: Resultado do comando nmap -sA ao servidor

Através da análise dos resultados obtidos no terminal, observamos que nem no caso da máquina virtual, nem caso do servidor existe uma firewall a filtrar alguma das 1000 portas.

Pode observar-se o tráfego no wireshark após o comando nmap -sA 172.16.7.2:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	PcsCompu_91:80:6a	Broadcast	ARP	42	Who has 172.16.7.2? Tell 172.16.7.1
2	0.000325011	PcsCompu_ed:da:41	PcsCompu_91:80:6a	ARP	60	172.16.7.2 is at 08:00:27:ed:da:41
3	0.017382734	172.16.7.1	172.16.7.2	TCP	54	35964 → 21 [ACK] Seq=1 Ack=1 Win=1024 Len=0
4	0.017438081	172.16.7.1	172.16.7.2	TCP	54	35964 → 143 [ACK] Seq=1 Ack=1 Win=1024 Len=0
5	0.017463145	172.16.7.1	172.16.7.2	TCP	54	35964 → 113 [ACK] Seq=1 Ack=1 Win=1024 Len=0
6	0.017486619	172.16.7.1	172.16.7.2	TCP	54	35964 → 22 [ACK] Seq=1 Ack=1 Win=1024 Len=0
7	0.017510743	172.16.7.1	172.16.7.2	TCP	54	35964 → 443 [ACK] Seq=1 Ack=1 Win=1024 Len=0
8	0.017535924	172.16.7.1	172.16.7.2	TCP	54	35964 → 23 [ACK] Seq=1 Ack=1 Win=1024 Len=0
9	0.017561302	172.16.7.1	172.16.7.2	TCP	54	35964 → 1025 [ACK] Seq=1 Ack=1 Win=1024 Len=0
10	0.017585463	172.16.7.1	172.16.7.2	TCP	54	35964 → 8888 [ACK] Seq=1 Ack=1 Win=1024 Len=0
11	0.017614856	172.16.7.1	172.16.7.2	TCP	54	35964 → 1723 [ACK] Seq=1 Ack=1 Win=1024 Len=0
12	0.017652235	172.16.7.2	172.16.7.1	TCP	60	21 → 35964 [RST] Seq=0 Win=0 Len=0
13	0.017699592	172.16.7.1	172.16.7.2	TCP	54	35964 → 8080 [ACK] Seq=1 Ack=1 Win=1024 Len=0
14	0.017723890	172.16.7.2	172.16.7.1	TCP	60	143 → 35964 [RST] Seq=1 Win=0 Len=0
15	0.017790899	172.16.7.2	172.16.7.1	TCP	60	113 → 35964 [RST] Seq=1 Win=0 Len=0
16	0.017794930	172.16.7.2	172.16.7.1	TCP	60	22 → 35964 [RST] Seq=1 Win=0 Len=0
17	0.017884698	172.16.7.2	172.16.7.1	TCP	60	443 → 35964 [RST] Seq=1 Win=0 Len=0
18	0.017889800	172.16.7.2	172.16.7.1	TCP	60	23 → 35964 [RST] Seq=1 Win=0 Len=0
19	0.017891607	172.16.7.2	172.16.7.1	TCP	60	1825 → 35964 [RST] Seq=1 Win=0 Len=0
20	0.017893409	172.16.7.2	172.16.7.1	TCP	60	8888 → 35964 [RST] Seq=1 Win=0 Len=0
21	0.017997364	172.16.7.2	172.16.7.1	TCP	60	1723 → 35964 [RST] Seq=1 Win=0 Len=0
22	0.018002427	172.16.7.2	172.16.7.1	TCP	60	8080 → 35964 [RST] Seq=1 Win=0 Len=0
23	0.018053430	172.16.7.1	172.16.7.2	TCP	54	35964 → 135 [ACK] Seq=1 Ack=1 Win=1024 Len=0
24	0.018083134	172.16.7.1	172.16.7.2	TCP	54	35964 → 554 [ACK] Seq=1 Ack=1 Win=1024 Len=0
25	0.018107591	172.16.7.1	172.16.7.2	TCP	54	35964 → 199 [ACK] Seq=1 Ack=1 Win=1024 Len=0
26	0.018130601	172.16.7.1	172.16.7.2	TCP	54	35964 → 111 [ACK] Seq=1 Ack=1 Win=1024 Len=0
27	0.018153584	172.16.7.1	172.16.7.2	TCP	54	35964 → 993 [ACK] Seq=1 Ack=1 Win=1024 Len=0
28	0.018176500	172.16.7.1	172.16.7.2	TCP	54	35964 → 110 [ACK] Seq=1 Ack=1 Win=1024 Len=0

Figura 2.13: Resultado do comando nmap -sA ao metasploitable no Wireshark

Pode observar-se o tráfego no wireshark após o comando nmap -sA 45.33.32.156:

No.	Time	Source	Destination	Protocol	Length	Info
4 0.000230472	10.0.2.15	45.33.32.156	45.33.32.156	ICMP	54	Timestamp request id=0x787b, seq=0/0, ttl=43
5 0.000430145	45.33.32.156	10.0.2.15	10.0.2.15	TCP	60	80 -- 42943 [RST] Seq=1 Win=0 Len=0
6 0.004692817	10.0.2.15	192.168.1.254	192.168.1.254	DNS	85	Standard query 0x1277 PTR 156.32.33.45.in-addr.arpa
7 0.016846401	192.168.1.254	10.0.2.15	10.0.2.15	DNS	434	Standard query response 0x1277 PTR 156.32.33.45.in-addr.arpa PTR s...
8 0.017416101	10.0.2.15	45.33.32.156	45.33.32.156	TCP	54	43199 -- 23 [ACK] Seq=1 Ack=1 Win=1024 Len=0
9 0.017511610	10.0.2.15	45.33.32.156	45.33.32.156	TCP	54	43199 -- 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0
10 0.017565928	45.33.32.156	10.0.2.15	10.0.2.15	TCP	60	23 -- 43199 [RST] Seq=1 Win=0 Len=0
11 0.017614767	10.0.2.15	45.33.32.156	45.33.32.156	TCP	54	43199 -- 993 [ACK] Seq=1 Ack=1 Win=1024 Len=0
12 0.017660443	45.33.32.156	10.0.2.15	10.0.2.15	TCP	60	80 -- 43199 [RST] Seq=1 Win=0 Len=0
13 0.017694345	10.0.2.15	45.33.32.156	45.33.32.156	TCP	54	43199 -- 1723 [ACK] Seq=1 Ack=1 Win=1024 Len=0
14 0.017742480	45.33.32.156	10.0.2.15	10.0.2.15	TCP	60	993 -- 43199 [RST] Seq=1 Win=0 Len=0
15 0.017777860	10.0.2.15	45.33.32.156	45.33.32.156	TCP	54	43199 -- 443 [ACK] Seq=1 Ack=1 Win=1024 Len=0
16 0.017834769	45.33.32.156	10.0.2.15	10.0.2.15	TCP	60	1723 -- 43199 [RST] Seq=1 Win=0 Len=0
17 0.017839089	45.33.32.156	10.0.2.15	10.0.2.15	TCP	60	443 -- 43199 [RST] Seq=1 Win=0 Len=0
18 0.017876730	10.0.2.15	45.33.32.156	45.33.32.156	TCP	54	43199 -- 199 [ACK] Seq=1 Ack=1 Win=1024 Len=0
19 0.017923563	45.33.32.156	10.0.2.15	10.0.2.15	TCP	60	199 -- 43199 [RST] Seq=1 Win=0 Len=0
20 0.017957468	10.0.2.15	45.33.32.156	45.33.32.156	TCP	54	43199 -- 21 [ACK] Seq=1 Ack=1 Win=1024 Len=0
21 0.018008899	10.0.2.15	45.33.32.156	45.33.32.156	TCP	54	43199 -- 139 [ACK] Seq=1 Ack=1 Win=1024 Len=0
22 0.018050819	45.33.32.156	10.0.2.15	10.0.2.15	TCP	60	21 -- 43199 [RST] Seq=1 Win=0 Len=0
23 0.018083914	10.0.2.15	45.33.32.156	45.33.32.156	TCP	54	43199 -- 113 [ACK] Seq=1 Ack=1 Win=1024 Len=0
24 0.018126650	45.33.32.156	10.0.2.15	10.0.2.15	TCP	60	139 -- 43199 [RST] Seq=1 Win=0 Len=0
25 0.018159367	10.0.2.15	45.33.32.156	45.33.32.156	TCP	54	43199 -- 256 [ACK] Seq=1 Ack=1 Win=1024 Len=0
26 0.018208574	45.33.32.156	10.0.2.15	10.0.2.15	TCP	60	113 -- 43199 [RST] Seq=1 Win=0 Len=0
27 0.018212196	45.33.32.156	10.0.2.15	10.0.2.15	TCP	60	256 -- 43199 [RST] Seq=1 Win=0 Len=0
28 0.018280604	10.0.2.15	45.33.32.156	45.33.32.156	TCP	54	43199 -- 554 [ACK] Seq=1 Ack=1 Win=1024 Len=0
29 0.018334781	10.0.2.15	45.33.32.156	45.33.32.156	TCP	54	43199 -- 3306 [ACK] Seq=1 Ack=1 Win=1024 Len=0

Figura 2.14: Resultado do comando nmap -sA ao servidor no Wireshark

Em ambas as figuras, observa-se que para cada pacote do tipo *ACK* enviado, é recebido um pacote do tipo *RST*. Com este comportamento, verifica-se como em cima descrito, que as portas não estão a ser filtradas.

2) nmap -sF / -sN / -sX target

Nesta questão foram criados 3 tipos de scans, devido ao elevado número de firewalls e IDS que procura por SYN scans. Cada um destes scans criados refere-se às flags de um TCP header. Neste tipo de scan, pretende-se que um port fechado responda com um RST na receção de pacotes. Num port aberto deve ser feito o drop dos pacotes pois estes ports aguardam por pacotes com SYN. Desta forma, uma conexão nunca é realizada e nem é enviado um pacote SYN.

⇒ nmap -sF target

Este tipo de scan é o FIN scan responsável por enviar um pacote TCP com a flag FIN ativa.

```

root@kali:~# nmap -sF 172.16.7.2
Starting Nmap 7.80 (@ https://nmap.org ) at 2019-11-29 23:58 WET
Nmap scan report for 172.16.7.2
Host is up (0.00018s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open|filtered  ftp
22/tcp    open|filtered  ssh
23/tcp    open|filtered  telnet
25/tcp    open|filtered  smtp
53/tcp    open|filtered  domain
80/tcp    open|filtered  http
111/tcp   open|filtered  rpcbind
139/tcp   open|filtered  netbios-ssn
445/tcp   open|filtered  microsoft-ds
512/tcp   open|filtered  exec
513/tcp   open|filtered  login
514/tcp   open|filtered  shell
1099/tcp  open|filtered  rmiregistry
1524/tcp  open|filtered  ingreslock
2049/tcp  open|filtered  nfs
2121/tcp  open|filtered  ccproxy-ftp
3306/tcp  open|filtered  mysql
5432/tcp  open|filtered  postgresql
5900/tcp  open|filtered  vnc
6000/tcp  open|filtered  X11
6667/tcp  open|filtered  irc
8009/tcp  open|filtered  ajp13
8180/tcp  open|filtered  unknown
MAC Address: 08:00:27:ED:DA:41 (Oracle VirtualBox virtual NIC)
ame 1996: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on
Nmap done: 1 IP address (1 host up) scanned in 1.44 seconds

```

Figura 2.15: Resultado do comando nmap -sF ao metasploitable

```

root@kali:~# nmap -sF 45.33.32.156      45.33.32.156      TCP
Starting Nmap 7.80 (@ https://nmap.org ) at 2019-11-29 23:59 WET
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.000042s latency).
All 1000 scanned ports on scanme.nmap.org (45.33.32.156) are closed
ame 1988: 54 bytes on wire (432 bits), 54 bytes captured (432 bits)
Nmap done: 1 IP address (1 host up) scanned in 0.16 seconds

```

Figura 2.16: Resultado do comando nmap -sF ao servidor

Com a utilização da flag -sF observa-se que na máquina virtual foram encontradas várias portas abertas com diferentes serviços em execução. No entanto, no servidor, todas as portas estão fechadas.

No.	Time	Source	Destination	Protocol	Length	Info
1 0.0000000000		PcsCompu_91:80:6a	Broadcast	ARP	42	Who has 172.16.7.2? Tell 172.16.7.1
2 0.000328757		PcsCompu_ed:da:41	PcsCompu_91:80:6a	ARP	60	172.16.7.2 is at 08:00:27:ed:da:41
3 0.035328728	172.16.7.1		172.16.7.2	TCP	54	59911 → 587 [FIN] Seq=1 Win=1024 Len=0
4 0.035531410	172.16.7.1		172.16.7.2	TCP	54	59911 → 23 [FIN] Seq=1 Win=1024 Len=0
5 0.035694694	172.16.7.1		172.16.7.2	TCP	54	59911 → 445 [FIN] Seq=1 Win=1024 Len=0
6 0.035721433	172.16.7.2		172.16.7.1	TCP	60	587 → 59911 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
7 0.035777280	172.16.7.1		172.16.7.2	TCP	54	59911 → 199 [FIN] Seq=1 Win=1024 Len=0
8 0.035835682	172.16.7.1		172.16.7.2	TCP	54	59911 → 995 [FIN] Seq=1 Win=1024 Len=0
9 0.035861565	172.16.7.1		172.16.7.2	TCP	54	59911 → 1720 [FIN] Seq=1 Win=1024 Len=0
10 0.035886623	172.16.7.1		172.16.7.2	TCP	54	59911 → 143 [FIN] Seq=1 Win=1024 Len=0
11 0.035910745	172.16.7.1		172.16.7.2	TCP	54	59911 → 3306 [FIN] Seq=1 Win=1024 Len=0
12 0.035941244	172.16.7.1		172.16.7.2	TCP	54	59911 → 80 [FIN] Seq=1 Win=1024 Len=0
13 0.035982748	172.16.7.2		172.16.7.1	TCP	60	199 → 59911 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
14 0.035987680	172.16.7.2		172.16.7.1	TCP	60	995 → 59911 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
15 0.036026840	172.16.7.1		172.16.7.2	TCP	54	59911 → 443 [FIN] Seq=1 Win=1024 Len=0
16 0.036054996	172.16.7.2		172.16.7.1	TCP	60	1720 → 59911 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
17 0.0360658947	172.16.7.2		172.16.7.1	TCP	60	143 → 59911 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
18 0.036215517	172.16.7.2		172.16.7.1	TCP	60	443 → 59911 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
19 0.038530084	172.16.7.1		172.16.7.2	TCP	54	59911 → 554 [FIN] Seq=1 Win=1024 Len=0
20 0.038575457	172.16.7.1		172.16.7.2	TCP	54	59911 → 8080 [FIN] Seq=1 Win=1024 Len=0
21 0.038601225	172.16.7.1		172.16.7.2	TCP	54	59911 → 1025 [FIN] Seq=1 Win=1024 Len=0
22 0.038624657	172.16.7.1		172.16.7.2	TCP	54	59911 → 22 [FIN] Seq=1 Win=1024 Len=0
23 0.038647942	172.16.7.1		172.16.7.2	TCP	54	59911 → 25 [FIN] Seq=1 Win=1024 Len=0
24 0.038672557	172.16.7.1		172.16.7.2	TCP	54	59911 → 111 [FIN] Seq=1 Win=1024 Len=0
25 0.038696435	172.16.7.1		172.16.7.2	TCP	54	59911 → 21 [FIN] Seq=1 Win=1024 Len=0
26 0.038720793	172.16.7.1		172.16.7.2	TCP	54	59911 → 139 [FIN] Seq=1 Win=1024 Len=0
27 0.038744338	172.16.7.2		172.16.7.1	TCP	60	554 → 59911 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
28 0.038793471	172.16.7.1		172.16.7.2	TCP	54	59911 → 256 [FIN] Seq=1 Win=1024 Len=0

Figura 2.17: Resultado do comando nmap -sF ao metasploitable no wireshark

No.	Time	Source	Destination	Protocol	Length	Info
5 0.002276540		45.33.32.156	10.0.2.15	TCP	60	80 → 62324 [RST, ACK] Seq=1 Win=0 Len=0
6 0.002734764	10.0.2.15		192.168.1.254	DNS	85	Standard query 0xb283 PTR 156.32.33.45.in-addr.arpa
7 0.019898943	192.168.1.254		10.0.2.15	DNS	434	Standard query response 0xb283 PTR 156.32.33.45.in-addr.arpa PTR s...
8 0.020483804	10.0.2.15		45.33.32.156	TCP	54	62580 → 256 [FIN] Seq=1 Win=1024 Len=0
9 0.020584565	10.0.2.15		45.33.32.156	TCP	54	62580 → 5900 [FIN] Seq=1 Win=1024 Len=0
10 0.020651433	45.33.32.156		10.0.2.15	TCP	60	256 → 62580 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
11 0.020693019	10.0.2.15		45.33.32.156	TCP	54	62580 → 23 [FIN] Seq=1 Win=1024 Len=0
12 0.020738536	45.33.32.156		10.0.2.15	TCP	60	5900 → 62580 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
13 0.020773394	10.0.2.15		45.33.32.156	TCP	54	62580 → 443 [FIN] Seq=1 Win=1024 Len=0
14 0.020816536	45.33.32.156		10.0.2.15	TCP	60	23 → 62580 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
15 0.020820071	45.33.32.156		10.0.2.15	TCP	60	443 → 62580 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
16 0.020853084	10.0.2.15		45.33.32.156	TCP	54	62580 → 1025 [FIN] Seq=1 Win=1024 Len=0
17 0.020904611	10.0.2.15		45.33.32.156	TCP	54	62580 → 8888 [FIN] Seq=1 Win=1024 Len=0
18 0.020948302	45.33.32.156		10.0.2.15	TCP	60	1025 → 62580 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
19 0.020952140	45.33.32.156		10.0.2.15	TCP	60	8888 → 62580 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
20 0.020984799	10.0.2.15		45.33.32.156	TCP	54	62580 → 554 [FIN] Seq=1 Win=1024 Len=0
21 0.021037011	10.0.2.15		45.33.32.156	TCP	54	62580 → 993 [FIN] Seq=1 Win=1024 Len=0
22 0.021087754	10.0.2.15		45.33.32.156	TCP	54	62580 → 80 [FIN] Seq=1 Win=1024 Len=0
23 0.021130580	45.33.32.156		10.0.2.15	TCP	60	554 → 62580 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
24 0.021134018	45.33.32.156		10.0.2.15	TCP	60	993 → 62580 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
25 0.021167137	10.0.2.15		45.33.32.156	TCP	54	62580 → 995 [FIN] Seq=1 Win=1024 Len=0
26 0.021210152	45.33.32.156		10.0.2.15	TCP	60	80 → 62580 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
27 0.021317945	45.33.32.156		10.0.2.15	TCP	60	995 → 62580 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
28 0.021369955	10.0.2.15		45.33.32.156	TCP	54	62580 → 111 [FIN] Seq=1 Win=1024 Len=0
29 0.021425755	10.0.2.15		45.33.32.156	TCP	54	62580 → 110 [FIN] Seq=1 Win=1024 Len=0
30 0.021471082	45.33.32.156		10.0.2.15	TCP	60	111 → 62580 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
31 0.021474867	45.33.32.156		10.0.2.15	TCP	60	110 → 62580 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
32 0.021508404	10.0.2.15		45.33.32.156	TCP	54	62580 → 3306 [FIN] Seq=1 Win=1024 Len=0

Figura 2.18: Resultado do comando nmap -sF ao servidor no wireshark

Como é observado nas imagens capturadas no Wireshark, existem diversos pacotes TCP "vazios" a serem enviados e caso a porta esteja fechada, para cada pacote vazio enviado recebemos imediatamente um pacote do tipo RST contendo apenas a flag FYN.

⇒ nmap -sN target

Este tipo de scan é o Null scan responsável por enviar um pacote TCP com a TCP flag header a 0.

```

root@kali:~# nmap -sN 172.16.7.2
Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-30 00:01 WET
Nmap scan report for 172.16.7.2
Host is up (0.00018s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open|filtered  ftp
22/tcp    open|filtered  ssh
23/tcp    open|filtered  telnet
25/tcp    open|filtered  smtp
53/tcp    open|filtered  domain
80/tcp    open|filtered  http
111/tcp   open|filtered  rpcbind
139/tcp   open|filtered  netbios-ssn
445/tcp   open|filtered  microsoft-ds
512/tcp   open|filtered  exec
513/tcp   open|filtered  login
514/tcp   open|filtered  shell
1099/tcp  open|filtered  rmiregistry
1524/tcp  open|filtered  ingreslock
2049/tcp  open|filtered  nfs
2121/tcp  open|filtered  ccproxy-ftp
3306/tcp  open|filtered  mysql
5432/tcp  open|filtered  postgresql
5900/tcp  open|filtered  vnc
6000/tcp  open|filtered  X11
6667/tcp  open|filtered  irc
8009/tcp  open|filtered  ajp13
8180/tcp  open|filtered  unknown
MAC Address: 08:00:27:ED:DA:41 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 1.32 seconds

```

Figura 2.19: Resultado do comando nmap -sN ao metasploitable

```

root@kali:~# nmap -sN 45.33.32.156
Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-30 00:03 WET
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.00017s latency).
All 1000 scanned ports on scanme.nmap.org (45.33.32.156) are closed
Nmap done: 1 IP address (1 host up) scanned in 0.16 seconds

```

Figura 2.20: Resultado do comando nmap -sN ao servidor

É possível observar que ao utilizar a flag -sN, percebemos que existem, na máquina virtual, algumas portas que poderão estar abertas/filtradas e identificamos os serviços que correm nessas portas. No servidor, todas as portas encontram-se fechadas.

No.	Time	Source	Destination	Protocol	Length	Info
9 0.028016941	172.16.7.1	172.16.7.2	TCP	54	56444 → 995 [None] Seq=1 Win=1024 Len=0	
10 0.028056062	172.16.7.2	172.16.7.1	TCP	60	1925 → 56444 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
11 0.028059191	172.16.7.2	172.16.7.1	TCP	60	8080 → 56444 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
12 0.028089981	172.16.7.1	172.16.7.2	TCP	54	56444 → 22 [None] Seq=1 Win=1024 Len=0	
13 0.028115293	172.16.7.1	172.16.7.2	TCP	54	56444 → 3389 [None] Seq=1 Win=1024 Len=0	
14 0.028137862	172.16.7.1	172.16.7.2	TCP	54	56444 → 110 [None] Seq=1 Win=1024 Len=0	
15 0.028167574	172.16.7.2	172.16.7.1	TCP	60	995 → 56444 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
16 0.028202268	172.16.7.1	172.16.7.2	TCP	54	56444 → 993 [None] Seq=1 Win=1024 Len=0	
17 0.028239579	172.16.7.2	172.16.7.1	TCP	60	3389 → 56444 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
18 0.028242396	172.16.7.2	172.16.7.1	TCP	60	110 → 56444 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
19 0.028437709	172.16.7.2	172.16.7.1	TCP	60	993 → 56444 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
20 0.030781728	172.16.7.1	172.16.7.2	TCP	54	56444 → 256 [None] Seq=1 Win=1024 Len=0	
21 0.030826463	172.16.7.1	172.16.7.2	TCP	54	56444 → 8888 [None] Seq=1 Win=1024 Len=0	
22 0.030851691	172.16.7.1	172.16.7.2	TCP	54	56444 → 445 [None] Seq=1 Win=1024 Len=0	
23 0.030874245	172.16.7.1	172.16.7.2	TCP	54	56444 → 554 [None] Seq=1 Win=1024 Len=0	
24 0.030896214	172.16.7.1	172.16.7.2	TCP	54	56444 → 143 [None] Seq=1 Win=1024 Len=0	
25 0.030919530	172.16.7.1	172.16.7.2	TCP	54	56444 → 23 [None] Seq=1 Win=1024 Len=0	
26 0.030941891	172.16.7.1	172.16.7.2	TCP	54	56444 → 139 [None] Seq=1 Win=1024 Len=0	
27 0.030964393	172.16.7.1	172.16.7.2	TCP	54	56444 → 587 [None] Seq=1 Win=1024 Len=0	
28 0.030987555	172.16.7.1	172.16.7.2	TCP	54	56444 → 135 [None] Seq=1 Win=1024 Len=0	
29 0.031016016	172.16.7.1	172.16.7.2	TCP	54	56444 → 1723 [None] Seq=1 Win=1024 Len=0	
30 0.031044508	172.16.7.2	172.16.7.1	TCP	60	256 → 56444 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
31 0.031051079	172.16.7.2	172.16.7.1	TCP	60	8888 → 56444 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
32 0.031094838	172.16.7.1	172.16.7.2	TCP	54	56444 → 3306 [None] Seq=1 Win=1024 Len=0	
33 0.031145603	172.16.7.2	172.16.7.1	TCP	60	554 → 56444 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
34 0.031149795	172.16.7.2	172.16.7.1	TCP	60	143 → 56444 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
35 0.031189128	172.16.7.1	172.16.7.2	TCP	54	56444 → 53 [None] Seq=1 Win=1024 Len=0	
36 0.031229904	172.16.7.2	172.16.7.1	TCP	60	587 → 56444 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	

Figura 2.21: Resultado do comando nmap -sN ao metasploitable no wireshark

No.	Time	Source	Destination	Protocol	Length	Info
9 0.018996836	10.0.2.15	45.33.32.156	TCP	54	42600 → 256 [None] Seq=1 Win=1024 Len=0	
10 0.019051037	45.33.32.156	10.0.2.15	TCP	60	443 → 42600 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
11 0.019059197	45.33.32.156	10.0.2.15	TCP	60	256 → 42600 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
12 0.019093452	10.0.2.15	45.33.32.156	TCP	54	42600 → 22 [None] Seq=1 Win=1024 Len=0	
13 0.019143613	10.0.2.15	45.33.32.156	TCP	54	42600 → 587 [None] Seq=1 Win=1024 Len=0	
14 0.019183979	45.33.32.156	10.0.2.15	TCP	60	22 → 42600 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
15 0.019223241	10.0.2.15	45.33.32.156	TCP	54	42600 → 1025 [None] Seq=1 Win=1024 Len=0	
16 0.019263912	45.33.32.156	10.0.2.15	TCP	60	587 → 42600 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
17 0.019295880	10.0.2.15	45.33.32.156	TCP	54	42600 → 5900 [None] Seq=1 Win=1024 Len=0	
18 0.019355346	45.33.32.156	10.0.2.15	TCP	60	1925 → 42600 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
19 0.019359631	45.33.32.156	10.0.2.15	TCP	60	5900 → 42600 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
20 0.019394434	10.0.2.15	45.33.32.156	TCP	54	42600 → 53 [None] Seq=1 Win=1024 Len=0	
21 0.019443853	10.0.2.15	45.33.32.156	TCP	54	42600 → 135 [None] Seq=1 Win=1024 Len=0	
22 0.019492719	10.0.2.15	45.33.32.156	TCP	54	42600 → 143 [None] Seq=1 Win=1024 Len=0	
23 0.019582487	45.33.32.156	10.0.2.15	TCP	60	53 → 42600 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
24 0.019589026	45.33.32.156	10.0.2.15	TCP	60	135 → 42600 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
25 0.019590822	45.33.32.156	10.0.2.15	TCP	60	143 → 42600 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
26 0.019631463	10.0.2.15	45.33.32.156	TCP	54	42600 → 993 [None] Seq=1 Win=1024 Len=0	
27 0.019765761	45.33.32.156	10.0.2.15	TCP	60	993 → 42600 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
28 0.019820034	10.0.2.15	45.33.32.156	TCP	54	42600 → 113 [None] Seq=1 Win=1024 Len=0	
29 0.019873815	10.0.2.15	45.33.32.156	TCP	54	42600 → 554 [None] Seq=1 Win=1024 Len=0	
30 0.0199114270	45.33.32.156	10.0.2.15	TCP	60	113 → 42600 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
31 0.0199117791	45.33.32.156	10.0.2.15	TCP	60	554 → 42600 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0	
32 0.019948505	10.0.2.15	45.33.32.156	TCP	54	42600 → 23 [None] Seq=1 Win=1024 Len=0	

Figura 2.22: Resultado do comando nmap -sN ao servidor no wireshark

Como podemos constatar nas imagens em cima capturadas no wireshark, observamos que existem diversos pacotes TCP "vazios" a serem enviados e caso a porta esteja fechada, para cada pacote vazio enviado recebemos imediatamente um pacote do tipo RST. Assim, caso não se receba nenhum pacote de resposta, significa que a porta indicada está aberta ou filtrada.

```
⇒ nmap -sX target
```

Este tipo de scan é chamado de Xmas scan e é responsável por ligar as flags FIN, PSH e URG ,”iluminando o pacote como uma árvore de Natal”.

```
root@kali:~# nmap -sX 172.16.7.2
Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-30 00:04 WET
Nmap scan report for 172.16.7.2
Host is up (0.00013s latency).
Not shown: 4977 closed ports
PORT      STATE SERVICE
21/tcp    open|filtered  ftp
22/tcp    open|filtered  ssh
23/tcp    open|filtered  telnet
25/tcp    open|filtered  smtp
53/tcp    open|filtered  domain
80/tcp    open|filtered  http
111/tcp   open|filtered  rpcbind
139/tcp   open|filtered  netbios-ssn
445/tcp   open|filtered  microsoft-ds
512/tcp   open|filtered  exec
513/tcp   open|filtered  login
514/tcp   open|filtered  shell
1099/tcp  open|filtered  rmiregistry
1524/tcp  open|filtered  ingreslock
2049/tcp  open|filtered  nfs
2121/tcp  open|filtered  ccproxy-ftp
3306/tcp  open|filtered  mysql
5432/tcp  open|filtered  postgresql
5900/tcp  open|filtered  vnc
6000/tcp  open|filtered  X11
6667/tcp  open|filtered  irc
8009/tcp  open|filtered  ajp13
8180/tcp  open|filtered  unknown
MAC Address: 08:00:27:ED:DA:41 (Oracle VirtualBox virtual NIC)
ame 1997: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) 
Nmap done: 1 IP address (1 host up) scanned in 1.42 seconds
```

Figura 2.23: Resultado do comando nmap -sX ao metasploitable

```
root@kali:~# nmap -sX 45.33.32.156
Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-30 00:06 WET
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.000082s latency).
All 1000 scanned ports on scanme.nmap.org (45.33.32.156) are closed
ame 1997: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) 
Nmap done: 1 IP address (1 host up) scanned in 0.16 seconds
```

Figura 2.24: Resultado do comando nmap -sX ao servidor

Com a utilização da flash -sX, observamos que na máquina virtual todas as portas estão abertas e no servidor

todas estão fechadas.

No.	Time	Source	Destination	Protocol	Length	Info
9 0. 0.019315847	172.16.7.1	172.16.7.2		TCP	54	58241 → 587 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
10 0. 0.019339321	172.16.7.1	172.16.7.2		TCP	54	58241 → 8888 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
11 0. 0.019357506	172.16.7.2	172.16.7.1		TCP	60	1720 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
12 0. 0.019407783	172.16.7.1	172.16.7.2		TCP	54	58241 → 3389 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
13 0. 0.019453506	172.16.7.2	172.16.7.1		TCP	60	993 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
14 0. 0.019456873	172.16.7.2	172.16.7.1		TCP	60	110 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
15 0. 0.019458726	172.16.7.2	172.16.7.1		TCP	60	8080 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
16 0. 0.019493554	172.16.7.1	172.16.7.2		TCP	54	58241 → 1025 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
17 0. 0.019529693	172.16.7.2	172.16.7.1		TCP	60	587 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
18 0. 0.019532471	172.16.7.2	172.16.7.1		TCP	60	8888 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
19 0. 0.019534201	172.16.7.2	172.16.7.1		TCP	60	3389 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
20 0. 0.019698830	172.16.7.2	172.16.7.1		TCP	60	1025 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
21 0. 0.022176162	172.16.7.1	172.16.7.2		TCP	54	58241 → 113 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
22 0. 0.022222295	172.16.7.1	172.16.7.2		TCP	54	58241 → 443 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
23 0. 0.022247130	172.16.7.1	172.16.7.2		TCP	54	58241 → 111 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
24 0. 0.022270034	172.16.7.1	172.16.7.2		TCP	54	58241 → 139 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
25 0. 0.022292185	172.16.7.1	172.16.7.2		TCP	54	58241 → 23 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
26 0. 0.022316104	172.16.7.1	172.16.7.2		TCP	54	58241 → 3306 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
27 0. 0.022387129	172.16.7.1	172.16.7.2		TCP	54	58241 → 995 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
28 0. 0.022412775	172.16.7.2	172.16.7.1		TCP	60	113 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
29 0. 0.022418116	172.16.7.2	172.16.7.1		TCP	60	443 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
30 0. 0.022455578	172.16.7.1	172.16.7.2		TCP	54	58241 → 256 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
31 0. 0.022477085	172.16.7.2	172.16.7.1		TCP	60	995 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
32 0. 0.022513443	172.16.7.1	172.16.7.2		TCP	54	58241 → 554 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
33 0. 0.022565240	172.16.7.1	172.16.7.2		TCP	54	58241 → 199 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
34 0. 0.022585254	172.16.7.2	172.16.7.1		TCP	60	256 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
35 0. 0.022588090	172.16.7.2	172.16.7.1		TCP	60	554 → 58241 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0

Figura 2.25: Resultado do comando nmap -sX ao metasploitable no wireshark

No.	Time	Source	Destination	Protocol	Length	Info
53 0. 0.021875470	45.33.32.156	10.0.2.15		TCP	60	80 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
54 0. 0.021918875	10.0.2.15	45.33.32.156		TCP	54	43717 → 23 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
55 0. 0.021966747	45.33.32.156	10.0.2.15		TCP	60	23 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
56 0. 0.022004055	10.0.2.15	45.33.32.156		TCP	54	43717 → 111 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
57 0. 0.022051579	45.33.32.156	10.0.2.15		TCP	60	111 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
58 0. 0.022086641	10.0.2.15	45.33.32.156		TCP	54	43717 → 445 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
59 0. 0.022133068	45.33.32.156	10.0.2.15		TCP	60	445 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
60 0. 0.022167901	10.0.2.15	45.33.32.156		TCP	54	43717 → 3306 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
61 0. 0.022215858	45.33.32.156	10.0.2.15		TCP	60	3306 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
62 0. 0.0222560479	10.0.2.15	45.33.32.156		TCP	54	43717 → 53 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
63 0. 0.022296832	45.33.32.156	10.0.2.15		TCP	60	53 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
64 0. 0.022345325	10.0.2.15	45.33.32.156		TCP	54	43717 → 8008 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
65 0. 0.022391377	45.33.32.156	10.0.2.15		TCP	60	8008 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
66 0. 0.022424353	10.0.2.15	45.33.32.156		TCP	54	43717 → 1658 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
67 0. 0.022470122	45.33.32.156	10.0.2.15		TCP	60	1658 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
68 0. 0.022543960	10.0.2.15	45.33.32.156		TCP	54	43717 → 20221 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
69 0. 0.022589349	45.33.32.156	10.0.2.15		TCP	60	20221 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
70 0. 0.022621741	10.0.2.15	45.33.32.156		TCP	54	43717 → 2065 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
71 0. 0.022664903	45.33.32.156	10.0.2.15		TCP	60	2065 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
72 0. 0.022697925	10.0.2.15	45.33.32.156		TCP	54	43717 → 4567 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
73 0. 0.022740918	45.33.32.156	10.0.2.15		TCP	60	4567 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
74 0. 0.022773431	10.0.2.15	45.33.32.156		TCP	54	43717 → 31337 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
75 0. 0.022817571	45.33.32.156	10.0.2.15		TCP	60	31337 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
76 0. 0.022850141	10.0.2.15	45.33.32.156		TCP	54	43717 → 683 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
77 0. 0.022893426	45.33.32.156	10.0.2.15		TCP	60	683 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
78 0. 0.022926050	10.0.2.15	45.33.32.156		TCP	54	43717 → 57797 [FIN, PSH, URG] Seq=1 Win=1024 Urg=0 Len=0
79 0. 0.022970578	45.33.32.156	10.0.2.15		TCP	60	57797 → 43717 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Figura 2.26: Resultado do comando nmap -sX ao servidor no wireshark

Mais uma vez, a diferença entre os scans apresentados está nas flags ativas. Neste scan existem 3 flags ativas (FIN; PSH; URG) como podemos observar nas imagens da captura de tráfego do Wireshark.

2.0.3 Q3

O objetivo desta questão era descobrir o sistema operativo utilizado pela máquina Metasploitable 2. Para tal, foi utilizada a flag -A para fazer um scan com o nmap que é responsável por detetar o sistema operativo, nomeadamente a distribuição de linux do metasploitable.

```

root@kali:~# nmap -A 172.16.7.2
Starting Nmap 7.80 ( https://nmap.org ) at 2019-11-30 00:10 WET
Nmap scan report for 172.16.7.2
Host is up (0.00056s latency).

PORT      STATE SERVICE          VERSION
21/tcp    open  ftp              vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst: STAT: 0.022086641
|_ FTP server status: 
| Connected to 172.16.7.1
| 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
22/tcp    open  ssh              OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey: 2617141
|_ 1024 60:0f:cfc1:e0:c5:6a: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)
23/tcp    open  telnet           Linux telnetd
25/tcp    open  smtp             Postfix smtpd
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN,
|_ssl-date: 2019-11-29T21:14:07+00:00; -2h57m09s from scanner time.
| sslv2: 0 _022893426
|_SSLV2 supported
| ciphers: 22970578
|_SSL2_RC4_128_EXPORT40_WITH_MD5
|_SSL2_RC2_128_CBC_WITH_MD5 (480 bits), 60 bytes captured (480 bits) on interface 0
|_SSL2 DES_64_CBC_WITH_MD5
|_SSL2_RC4_128_WITH_MD5
|_SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
|_SSL2_DES_192_EDE3_CBC_WITH_MD5
53/tcp    open  domain            ISC BIND 9.4.2
| dns-nsid:
|_ bind.version: 9.4.2

```

Figura 2.27: Resultado do comando SO

```

80/tcp open  http   X Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
|_http-title: Metasploitable2 - Linux
111/tcp open  rpcbind  2 (RPC #100000)
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp open  exec    netkit-rsh rexecd
513/tcp open  login   OpenBSD or Solaris rlogin
514/tcp open  shell   Netkit rshd
1099/tcp open  java-rmi GNU Classpath grmiregistry
1524/tcp open  bindshell Metasploitable root shell
2049/tcp open  nfs    2-4 (RPC #100003)
2121/tcp open  ftp    ProFTPD 1.3.1
3306/tcp open  mysql MySQL 5.0.51a-Subuntu5
| mysql-info:
| Protocol: 10
| Version: 5.0.51a-Subuntu5
| Thread ID: 12
| Capabilities flags: 43564
| Some Capabilities: LongColumnFlag, SwitchToSSLAfterHandshake, SupportsTransactions, Speaks4ProtocolNew, Support41Auth, ConnectWithDatabase, Support
sCompression
| Status: Autocommit
| Salt: Qp`o;/xh#0;{xF4nR08e
5432/tcp open  postgresql PostgresSQL DB 8.3.0 - 8.3.7
| ssl-date: 2019-11-29T21:14:06+00:00; -2h57m09s from scanner time.
5900/tcp open  vnc   VNC (protocol 3.3)
| vnc-info:
| Protocol version: 3.3
| Security types:
|_ VNC Authentication (2)
6000/tcp open  x11   (access denied)
6667/tcp open  irc   UnrealIRCd
8009/tcp open  ajp13 Apache Jserv (Protocol v1.3)
|_ajp-methods: Failed to get a valid response for the OPTION (480 bits) on interface 0
8180/tcp open  http  Apache/Tomcat/Coyote JSP engine 1.1
|_http-favicon: Apache Tomcats 1b ab 2d 21 20 9c 0a 00  (FY---!----)
|_http-server-header: Apache-Coyote/1.1.5 aa 77 of 50 14  --2----f-w-P-
|_http-title: Apache Tomcat/5.5.06 0b 60 00 00
MAC Address: 08:00:27:ED:D4:41 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33

```

Figura 2.28: Resultado do comando SO

```

Network Distance: 1 hop
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
77.0.0.22893426 45.33.32.156 10.0.2.15 TCP 60 31337 - 43717 [FIN, PSH, URG] Seq=1 Ack=1 Win=0 Len=0
Host script results:
|_clock-skew: mean: -2h57m09s, deviation: 0s, median: -2h57m09s
|_ms-sql-info: ERROR: Script execution failed (use -d to debug)
|_nbstat: NetBIOS name: METASPOITABLE, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
|_smb-os-discovery: ERROR: Script execution failed (use -d to debug)
|_smb-security-mode:2ERROR: Script execution failed (use -d to debug)
|_smb2-time: Protocol negotiation failed (SMB2)9c 0a 00  (FY---!----)
0020 02 0f 04 32 aa c5 00 00 00 00 66 aa 77 ef 50 14  --2----f-w-P-
TRACEROUTE 00 c8 73 00 00 00 00 00 00 00 00 00 00 00 00 00
HOP RTT      ADDRESS
1  0.56 ms 172.16.7.2

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 36.49 seconds

```

Figura 2.29: Resultado do comando SO

2.0.4 Q4

Os três serviços que decidimos utilizar foram:

- ftp
- ssh
- smtp

As três vulnerabilidades mais recentes do ftp são:

CVE-2019-9809 Detail

Current Description

If the source for resources on a page is through an FTP connection, it is possible to trigger a series of modal alert messages for these resources through invalid credentials or locations. These messages cannot be immediately dismissed, allowing for a denial of service (DOS) attack. This vulnerability affects Firefox < 66.

Figura 2.30: vulnerabilidade 1-ftp

CVE-2019-9807 Detail

Current Description

When arbitrary text is sent over an FTP connection and a page reload is initiated, it is possible to create a modal alert message with this text as the content. This could potentially be used for social engineering attacks. This vulnerability affects Firefox < 66.

Figura 2.31: vulnerabilidade 2-ftp

CVE-2019-9806 Detail

Current Description

A vulnerability exists during authorization prompting for FTP transaction where successive modal prompts are displayed and cannot be immediately dismissed. This allows for a denial of service (DOS) attack. This vulnerability affects Firefox < 66.

Figura 2.32: vulnerabilidade 3-ftp

As três vulnerabilidades do ssh são:

CVE-2019-9228 Detail

Current Description

**** DISPUTED **** An issue was discovered on AudioCodes Mediant 500L-MSBR, 500-MBSR, M800B-MSBR and 800C-MSBR devices with firmware versions F7.20A at least to 7.20A.252.062. The (1) management SSH and (2) management TELNET features allow remote attackers to cause a denial of service (connection slot exhaustion) via 5 unauthenticated connection attempts, because the maximum number of unauthenticated clients that can be configured is 5. NOTE: the vendor's position is that this is a "design choice."

Figura 2.33: vulnerabilidade 1-ssh

CVE-2019-9160 Detail

Current Description

WAC on the Sangfor Sundray WLAN Controller version 3.7.4.2 and earlier has a backdoor account allowing a remote attacker to login to the system via SSH (on TCP port 22345) and escalate to root (because the password for root is the WebUI admin password concatenated with a static string).

Figura 2.34: vulnerabilidade 2-ssh

CVE-2019-7690 Detail

Current Description

In MobaTek MobaXterm Personal Edition v11.1 Build 3860, the SSH private key and its password can be retrieved from process memory for the lifetime of the process, even after the user disconnects from the remote SSH server. This affects Passwordless Authentication that has a Password Protected SSH Private Key.

Figura 2.35: vulnerabilidade 3-ssh

As vulnerabilidades do smtp são:

CVE-2019-9868 Detail

Current Description

An issue was discovered in the Web Console in Veritas NetBackup Appliance through 3.1.2. The SMTP password is displayed to an administrator.

Figura 2.36: vulnerabilidade 1-smtp

CVE-2019-6242 Detail

Current Description

**** DISPUTED **** Kentico v10.0.42 allows Global Administrators to read the cleartext SMTP Password by navigating to the SMTP configuration page. NOTE: the vendor considers this a best-practice violation but not a vulnerability. The vendor plans to fix it at a future time.

Figura 2.37: vulnerabilidade 2-smtp

CVE-2019-5456 Detail

MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

Current Description

SMTP MITM refers to a malicious actor setting up an SMTP proxy server between the UniFi Controller version <= 5.10.21 and their actual SMTP server to record their SMTP credentials for malicious use later.

Figura 2.38: vulnerabilidade 3-smtp

Capítulo 3

Parte 2

3.0.1 Q5

O principal objetivo desta pergunta é observar os resultados apresentados pelo Nessus do scanning ao VM *Metasploitable*. Concluímos que existem vários níveis de prioridade:

- **Critical:** Vulnerabilidades críticas ao sistema, por exemplo, quando um atacante altera ficheiros na máquina do utilizador.
- **High:** Vulnerabilidades de nível alto. Incluem remote service, aceitação de conexões encriptadas com SSL 2.0 e/ou SSL 3.0.
- **Medium:** Vulnerabilidades de nível médio, por exemplo, alguns métodos que se usam para fazer debug não devem estar ativos como é o caso de *HTTP Trace/ Track Methods Allowed*.
- **Low:** Vulnerabilidades de nível baixo.
- **Info:** O utilizador é informado dos serviços que podem ser considerados vulneráveis como é o caso de *Os identification*.

Para entendermos melhor as características e as vulnerabilidades da máquina virtual metaesploitable 2 foi feito um scan do tipo *advanced*. Concluímos então que não tem um ambiente seguro para uso pessoal.



Figura 3.1: Resultado do scan no Nessus

De seguida encontram-se alguns exemplos de cada grupo de vulnerabilidades:

<input type="checkbox"/>	CRITICAL	Bind Shell Backdoor Detection	Backdoors	1	
<input type="checkbox"/>	CRITICAL	NFS Exported Share Information Disclosure	RPC	1	
<input type="checkbox"/>	CRITICAL	rexecd Service Detection	Service detection	1	
<input type="checkbox"/>	CRITICAL	Unix Operating System Unsupported Vers...	General	1	

Figura 3.2: vulnerabilidades críticas

<input type="checkbox"/>	HIGH	rlogin Service Detection	Service detection	1	
--------------------------	------	--------------------------	-------------------	---	--

Figura 3.3: Vulnerabilidades altas

<input type="checkbox"/>	MEDIUM	NFS Shares World Readable	RPC	1	
<input type="checkbox"/>	MEDIUM	Samba Badlock Vulnerability	General	1	
<input type="checkbox"/>	MEDIUM	SMB Signing not required	Misc.	1	
<input type="checkbox"/>	MEDIUM	SSL DROWN Attack Vulnerability (Decryp...	Misc.	1	

Figura 3.4: Vulnerabilidades médias

<input type="checkbox"/>	LOW	SSL/TLS Diffie-Hellman Modulus <= 1024...	Misc.	1	
<input type="checkbox"/>	LOW	X Server Detection	Service detection	1	

Figura 3.5: Vulnerabilidades baixas

<input type="checkbox"/>	INFO	Nessus SYN scanner	Port scanners	25	
<input type="checkbox"/>	INFO	RPC Services Enumeration	Service detection	10	
<input type="checkbox"/>	INFO	Service Detection	Service detection	10	
<input type="checkbox"/>	INFO	SMB (Multiple Issues)	Windows	8	

Figura 3.6: Info sobre algumas possíveis vulnerabilidades

3.0.2 Q6

Para esta questão foi necessário examinar o output do Snort e escolher dois eventos considerados tráfego anómalo. Assim, acedemos aos logs do Snort e analisamos os mesmos. Desta forma, foram escolhidos eventos para analisar. No primeiro caso o Snort encontrou uma tentativa de Information Leak e no segundo foi detetada uma tentativa de Denial of Service.

```
[**] [1:1418:11] SNMP request tcp [**] [limi: 4656]          Service detection
[Classification: Attempted Information Leak] [Priority: 2]
12/04-13:36:04.019796 172.16.7.1:10821 -> 172.16.7.2:161   Vulnerabilities
TCP TTL:64 TOS:0x0 IPLen:20 DgmLen:48 DF                         Critical
*****S* Seq: 0xF275D4B2 Ack: 0x0 Win: 0x1000 TcpLen: 28
*****S* Seq: 0xF275D4B2 Ack: 0x0 Win: 0x1000 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK                         High
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2002-0013][Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2002-0012][Xref => http://www.se-                         Medium
curityfocus.com/bid/4132][Xref => http://www.securityfocus.com/bid/4089][Xref => http://www.securityfocus.com/bid/4088] d 1>
```

Figura 3.7: Primeira vulnerabilidade - Tentativa de Information Leak

```
[**] [1:249:8] DDOS mstream client to handler [**]
[Classification: Attempted Denial of Service] [Priority: 2]          Service detection
12/04-13:36:04.577308 172.16.7.1:12200 -> 172.16.7.2:15104      Snort!IDS111 -> http://www.whitehats.com/info/IDS111
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:48 DF
*****S* Seq: 0x2DA1A97A Ack: 0x0 Win: 0x1000 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2000-0138][Xref => http://www.whitehats.com/info/IDS111]
```

Figura 3.8: Segunda vulnerabilidade - Tentativa de Denial of Service

O tráfego em cima apresentado, considerado anómalo, foi capturado no Wireshark. Assim, é possível observar que os pacotes que se julgavam ser uma tentativa de ataque devido ao seu número elevado , conclui-se que são pacotes [SYN], em ambos os casos.

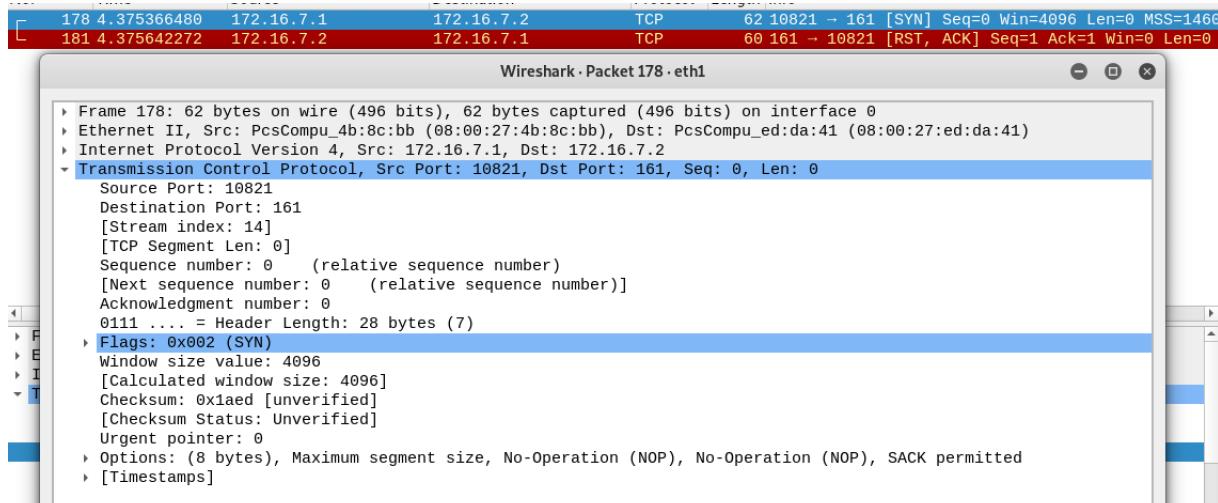


Figura 3.9: Tentativa de Information Leak no Wireshark

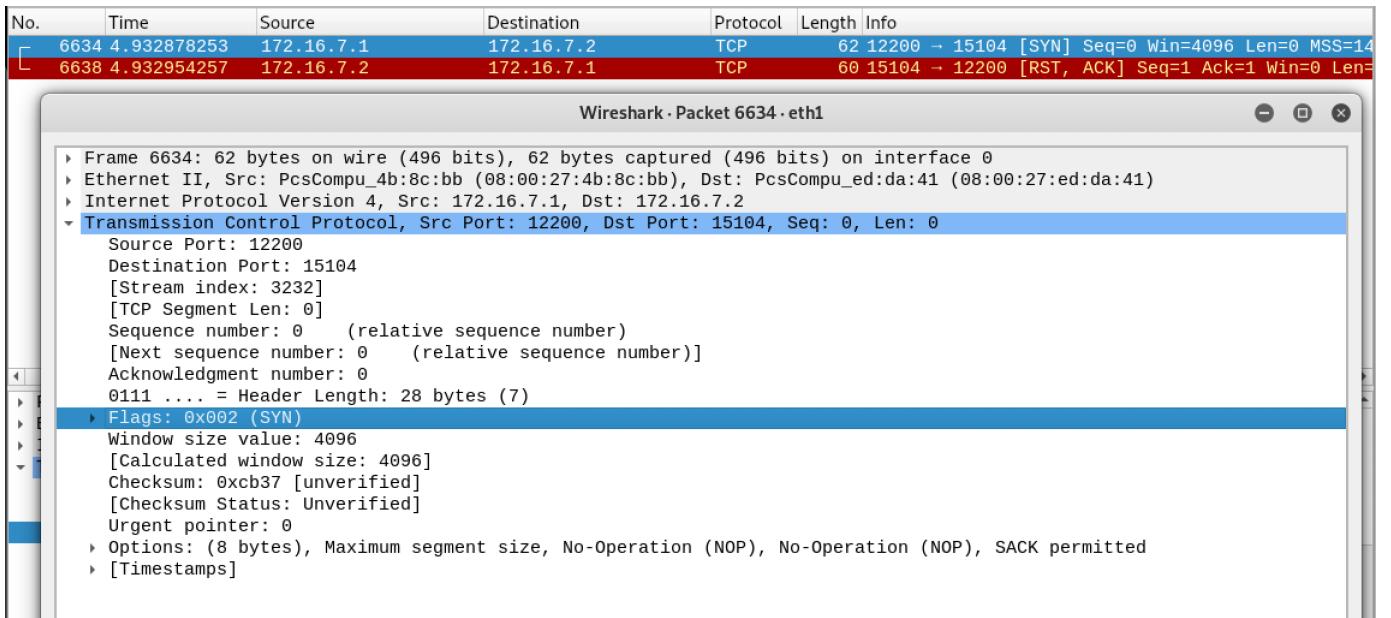


Figura 3.10: Tentativa de Deniel of Service no Wireshark

De seguida, são apresentados os CVE's das vulnerabilidades referidas em cima.

CVE-2002-0012 Detail

MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

Current Description

Vulnerabilities in a large number of SNMP implementations allow remote attackers to cause a denial of service or gain privileges via SNMPv1 trap handling, as demonstrated by the PROTOs c06-SNMPv1 test suite. NOTE: It is highly likely that this candidate will be SPLIT into multiple candidates, one or more for each vendor. This and other SNMP-related candidates will be updated when more accurate information is available.

Source: MITRE

[— Hide Analysis Description](#)

Analysis Description

Vulnerabilities in a large number of SNMP implementations allow remote attackers to cause a denial of service or gain privileges via SNMPv1 trap handling, as demonstrated by the PROTOs c06-SNMPv1 test suite. NOTE: It is highly likely that this candidate will be SPLIT into multiple candidates, one or more for each vendor. This and other SNMP-related candidates will be updated when more accurate information is available.

Source: MITRE

Figura 3.11: CVE da Information Leak

CVE-2000-0138 Detail

MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

Current Description

A system has a distributed denial of service (DDOS) attack master, agent, or zombie installed, such as (1) Trinoo, (2) Tribe Flood Network (TFN), (3) Tribe Flood Network 2000 (TFN2K), (4) stacheldraht, (5) mstream, or (6) shaft.

Source: MITRE

[— Hide Analysis Description](#)

Analysis Description

A system has a distributed denial of service (DDOS) attack master, agent, or zombie installed, such as (1) Trinoo, (2) Tribe Flood Network (TFN), (3) Tribe Flood Network 2000 (TFN2K), (4) stacheldraht, (5) mstream, or (6) shaft.

Source: MITRE

Figura 3.12: CVE da Denial of Service

3.0.3 Q7

The screenshot shows the 'RMI Registry Detection' plugin details page. It includes sections for Description, See Also, Output, Risk Information, and Vulnerability Information.

Description: The remote host is running an RMI registry, which acts as a bootstrap naming service for registering and retrieving remote objects with simple names in the Java Remote Method Invocation (RMI) system.

See Also: <https://docs.oracle.com/javase/1.5.0/docs/guide/rmi/spec/rmiTOC.html>
<http://www.nessus.org/u?b6fd7659>

Output: No output recorded.

Port ▲	Hosts
1099 / tcp / rmi_registry	172.16.7.2

Risk Information: Risk Factor: None

Vulnerability Information: CPE: cpe:/a:oracle:java_se
Asset Inventory: True

Figura 3.13: Com vulnerabilidade

The screenshot shows the 'RPC Services Enumeration' plugin details page. It includes sections for Description, Output, Risk Information, and Vulnerability Information.

Description: By sending a DUMP request to the portmapper, it was possible to enumerate the ONC RPC services running on the remote port. Using this information, it is possible to connect and bind to each service by sending an RPC request to the remote port.

Output: The following RPC services are available on TCP port 111 :
- program: 100000 (portmapper), version: 2

Port ▲	Hosts
111 / tcp / rpc-portma...	172.16.7.2

Risk Information: Risk Factor: None

Figura 3.14: Sem vulnerabilidade

Como se pode observar nas imagens acima apresentadas foram encontradas notificações do IDS que possuíam vulnerabilidades e outras que não. Tal se deve ao facto de, apesar de o Nessus executar estas verificações de vulnerabilidades nos serviços e de realizar o login para expor patches ausentes, a ausência de vulnerabilidades não quer dizer que o servidor esteja bem configurado. Saber como o servidor está configurado, as vulnerabilidades presentes e como os patches são usados permite reduzir os riscos.

A vantagem do uso do Nessus para detetar vulnerabilidades consiste no facto de os dados poderem ser analisados de uma só vez.

3.0.4 Q8

Esta questão tem como objetivo solucionar algumas das vulnerabilidades encontradas na VM Metaploitble 2. As vulnerabilidades que foram para corrigir são as seguintes:

- HTTP TRACK/TRACE Methods Allowed
- rexecd Service Detection
- Bind Shell Backdoor Detection
- VNC Server ‘password’ Password

Iremos então apresentar cada uma das vulnerabilidades referidas em cima e de que forma é que estas foram corrigidas.

HTTP TRACK/TRACE Methods Allowed

Esta vulnerabilidade indica que o servidor web remoto suporta os métodos TRACE e/ou TRACK. O TRACE e/ou TRACK são métodos HTTP utilizados para realizar o debug em conexões do web server. Pode realizar-se um ataque XSS quando combinado com outras vulnerabilidades.

De forma a solucionar, devem desligar-se estes métodos no ficheiro de configuração através da introduçao da linha TraceEnable Off.

MEDIUM **HTTP TRACE / TRACK Methods Allowed** »

Description
The remote web server supports the TRACE and/or TRACK methods. TRACE and TRACK are HTTP methods that are used to debug web server connections.

Solution
Disable these methods. Refer to the plugin output for more information.

Figura 3.15: Vulnerabilidade dos métodos HTTP TRACE/TRACK apresentada pelo Nessus

rexecd Service Detection

O rexecd possibilita aos utilizadores executarem comandos remotamente. Contudo, dado que não precisa de autenticação é um serviço vulnerável, estando propício a ataques. Para solucionar esta vunerabilidade devemos remover a entrada iniciada por um exec no ficheiro inetd.conf.

CRITICAL rexecd Service Detection < >

Description

The rexecd service is running on the remote host. This service is design to allow users of a network to execute commands remotely. However, rexecd does not provide any good means of authentication, so it may be abused by an attacker to scan a third-party host.

Solution

Comment out the 'exec' line in /etc/inetd.conf and restart the inetd process.

Figura 3.16: Vulnerabilidade rexecd Service Detention apresentada pelo Nessus

Bind Shell Backdoor Detection

A shell está a ouvir uma porta remota sem qualquer tipo de autenticação ter sido requerida. Um atacante deve usá-la através da conexão à port remota e enviando comandos diretamente. Como solução a esta vulnerabilidade deve verificar-se se o host remoto está comprometido e reinstalar o sistema caso seja necessário. Caso não tenha sido comprometido, podemos desativar o ficheiro de configuração do serviço que está a usar a port; podemos desativar a port 1524 e, por último, podemos filtrar a porta 1524 utilizando uma firewall para esse efeito.

CRITICAL Bind Shell Backdoor Detection < >

Description

A shell is listening on the remote port without any authentication being required. An attacker may use it by connecting to the remote port and sending commands directly.

Solution

Verify if the remote host has been compromised, and reinstall the system if necessary.

Figura 3.17: Vulnerabilidade da Bind Shell Backdoor Detection apresentada pelo Nessus

VNC Server ‘password’ Password

O servidor VNV está a correr num host remoto utilizando uma password fraca. O Nessus estava disponível para a realização do login através da utilização da autenticação VNC e a password ‘password’. Um atacante não autenticado pode explorar esta vulnerabilidade e tomar controlo do sistema.

Para solucionar esta vulnerabilidade devemos alterar a palavra passe de forma a que esta seja uma password forte. Assim, foi necessário descobrir o ficheiro de password na pasta root. Alteramos este ficheiro com o comando vncpasswd passwd. Normalmente o ficheiro encontra-se dentro da pasta.

CRITICAL VNC Server 'password' Password < >

Description

The VNC server running on the remote host is secured with a weak password. Nessus was able to login using VNC authentication and a password of 'password'. A remote, unauthenticated attacker could exploit this to take control of the system.

Solution

Secure the VNC service with a strong password.

Figura 3.18: Vulnerabilidade do servidor VNC apresentada no Nessus

Resultados Finais

Depois de ter sido realizada a correção das vulnerabilidades foi feito um novo scan sendo o resultado mostrado de seguida:

Sev	Name	Family	Count	Host Data
CRITICAL	SSL (Multiple Issues)	Gain a shell remotely	2	
CRITICAL	NFS Exported Share Issues	RPC	1	
CRITICAL	Unix Operating System Issues	General	1	
HIGH	SSL Version 2 and 3 Problems	Service detection	1	Modify
MIXED	SSL (Multiple Issues)	General	13	
MIXED	SSH (Multiple Issues)	Misc.	4	

Figura 3.19: Vulnerabilidades Corrigidas

Capítulo 4

Conclusão

A elaboração deste projeto, dado que se utilizou principalmente o Snort e o Nessus, ferramentas que o grupo nunca tinha contactado, permitiu-nos aprofundar conhecimentos e ter contacto no que toca a técnicas de PenTesting e identificação de vulnerabilidades através da deteção de tráfego anómalo.

Com o desenvolvimento deste trabalho conseguimos compreender como utilizar o Nmap e as várias vertentes deste. Utilizamos também o Wireshark para capturar tráfego (ferramenta já utilizada anteriormente pelo grupo).

Concluímos então este trabalho com satisfação dado os resultados obtidos bem como o facto de termos atingido os nossos objetivos.

Este projeto deu-nos uma maior noção e tornou-nos pessoas mais conscientes dado que teremos muito mais cuidado com as máquinas que iremos trabalhar pois encontramos várias vulnerabilidade que podem comprometer por completo o sistema no que toca à sua integridade, disponibilidade e confidencialidade.