Project II: Phishing Attacks in Wi-Fi Networks

Student ID: 0716085   Name: Pin-Hua Lai

Part0: Environment Setting

I choose ***scenario ii*** in this project.

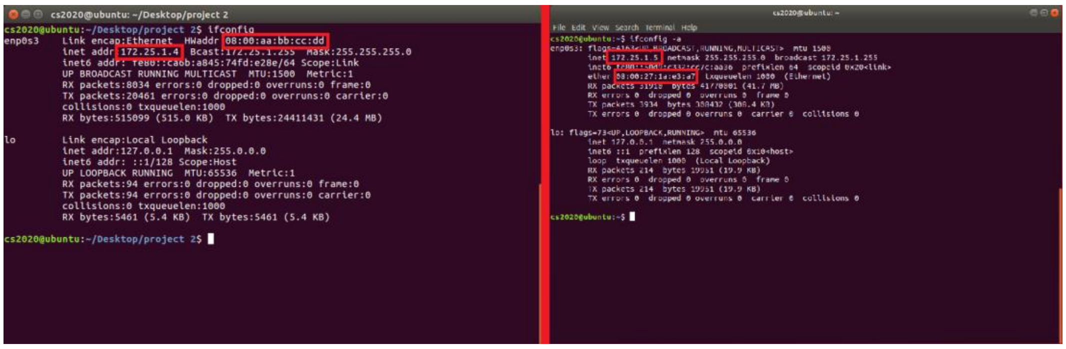
Scenario II (NAT Network):

  CIDR: 172.25.1.0/24

  AP: 172.25.1.1 MAC: 52:54:00:12:35:00

  Attacker: 172.25.1.4 MAC: 08:00:aa:bb:cc:dd

  Victim: IP: 172.25.1.5 MAC: 08:00:27:1a:e3:a7



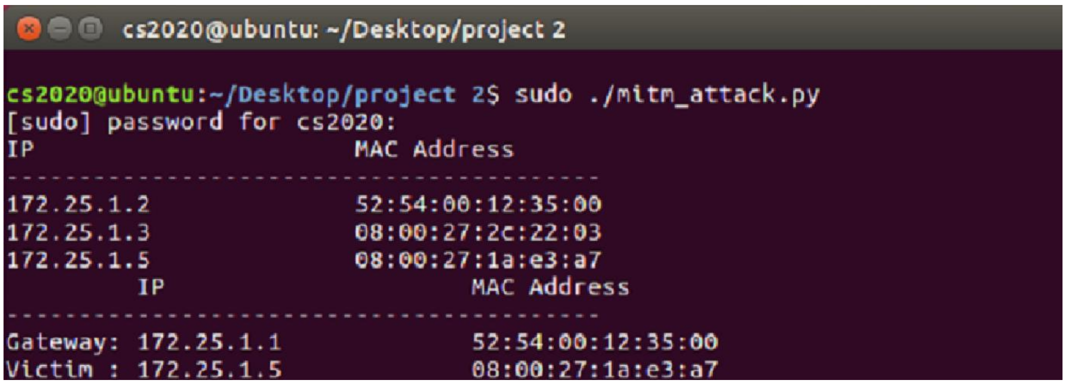
```
cs2020@ubuntu: ~/Desktop/project 2
cs2020@ubuntu:~/Desktop/project 2$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:aa:bb:cc:dd
          inet addr:172.25.1.4  Bcast:172.25.1.255  Mask:255.255.255.0
          inet6 addr: fe80::ca6b:a845:74fd:e28e/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:8034 errors:0 dropped:0 overruns:0 frame:0
          TX packets:20461 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:515099 (515.0 KB)  TX bytes:24411431 (24.4 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:94 errors:0 dropped:0 overruns:0 frame:0
          TX packets:94 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5461 (5.4 KB)  TX bytes:5461 (5.4 KB)

cs2020@ubuntu:~/Desktop/project 2$
```

```
cs2020@ubuntu: ~
File  Edit  View  Search  Terminal  Help
cs2020@ubuntu:~$ ifconfig -a
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.25.1.5  netmask 255.255.255.0  broadcast 172.25.1.255
        inet6 fe80::5b0d:c332:cc7c:aa36  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:1a:e3:a7  txqueuelen 1000  (Ethernet)
        RX packets 31918  bytes 41770801 (41.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3934  bytes 308432 (308.4 KB)
        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 214  bytes 19951 (19.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 214  bytes 19951 (19.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

cs2020@ubuntu:~$
```

(1) ifconfig (Attacker, Victim)

Part1: Man in the Middle attack

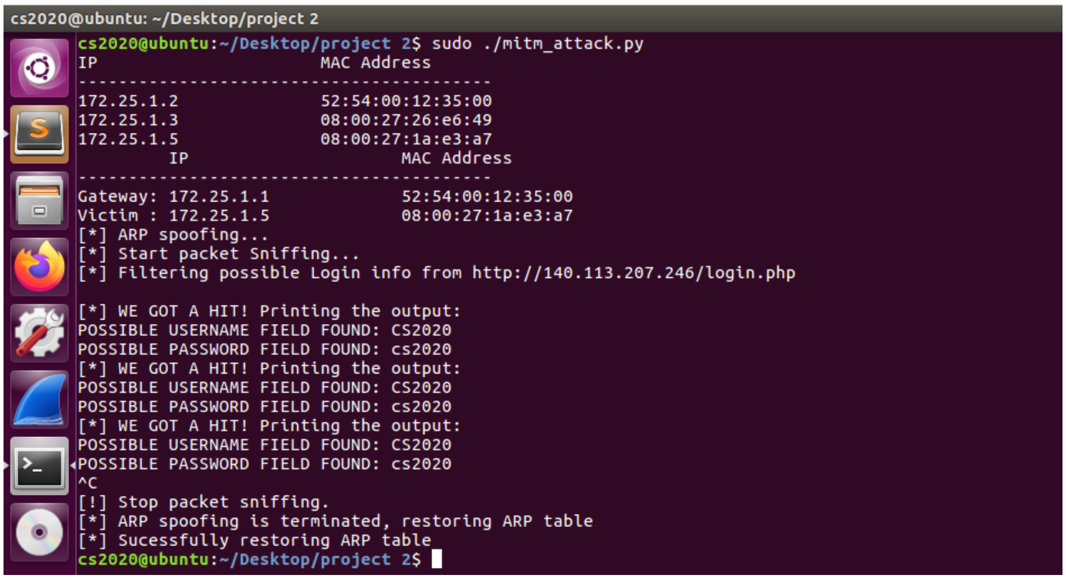
Task I: Obtain Other Client Devices' IP/MAC Addresses in a Wi-Fi Network



```
cs2020@ubuntu: ~/Desktop/project 2

cs2020@ubuntu:~/Desktop/project 2$ sudo ./mitm_attack.py
[sudo] password for cs2020:
IP                      MAC Address
-----------------------------------------
172.25.1.2              52:54:00:12:35:00
172.25.1.3              08:00:27:2c:22:03
172.25.1.5              08:00:27:1a:e3:a7
         IP                     MAC Address
-----------------------------------------
Gateway: 172.25.1.1             52:54:00:12:35:00
Victim : 172.25.1.5             08:00:27:1a:e3:a7
```

(1) Scanning result

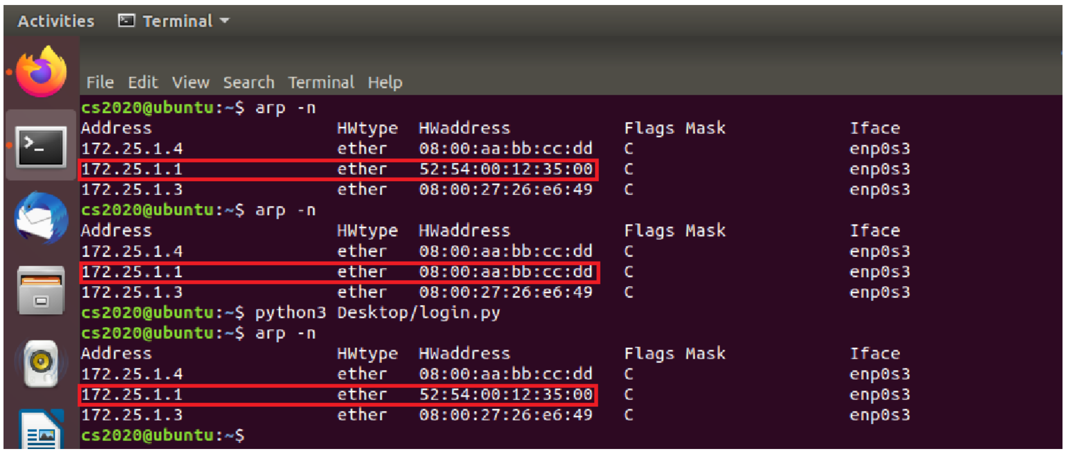
Task II: ARP Spoofing



```
cs2020@ubuntu:~/Desktop/project 2$ sudo ./mitm_attack.py
IP                        MAC Address
-----------------------------------------
172.25.1.2                52:54:00:12:35:00
172.25.1.3                08:00:27:26:e6:49
172.25.1.5                08:00:27:1a:e3:a7
         IP                       MAC Address
-----------------------------------------
Gateway: 172.25.1.1               52:54:00:12:35:00
Victim : 172.25.1.5               08:00:27:1a:e3:a7
[*] ARP spoofing...
[*] Start packet Sniffing...
[*] Filtering possible Login info from http://140.113.207.246/login.php

[*] WE GOT A HIT! Printing the output:
POSSIBLE USERNAME FIELD FOUND: CS2020
POSSIBLE PASSWORD FIELD FOUND: cs2020
[*] WE GOT A HIT! Printing the output:
POSSIBLE USERNAME FIELD FOUND: CS2020
POSSIBLE PASSWORD FIELD FOUND: cs2020
[*] WE GOT A HIT! Printing the output:
POSSIBLE USERNAME FIELD FOUND: CS2020
POSSIBLE PASSWORD FIELD FOUND: cs2020
^C
[!] Stop packet sniffing.
[*] ARP spoofing is terminated, restoring ARP table
[*] Sucessfully restoring ARP table
cs2020@ubuntu:~/Desktop/project 2$
```

(2) MitM attack at attacker side

Picture (2) shows the output of ***mitm_attack.py***, including the scanning result, the process and the possible login data by doing packet sniffing.



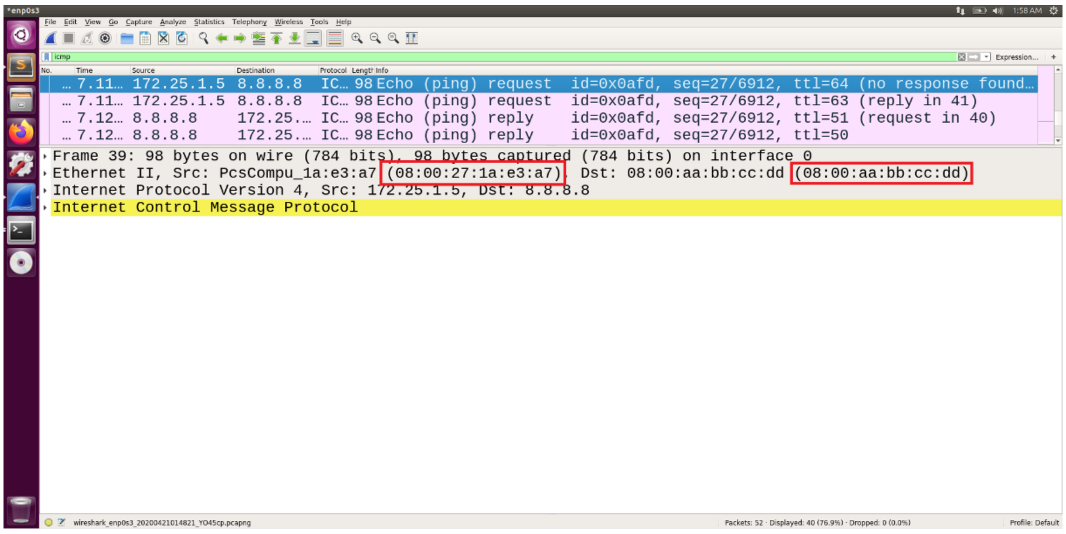
```
cs2020@ubuntu:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.25.1.4               ether   08:00:aa:bb:cc:dd   C                     enp0s3
172.25.1.1               ether   52:54:00:12:35:00   C                     enp0s3
172.25.1.3               ether   08:00:27:26:e6:49   C                     enp0s3
cs2020@ubuntu:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.25.1.4               ether   08:00:aa:bb:cc:dd   C                     enp0s3
172.25.1.1               ether   08:00:aa:bb:cc:dd   C                     enp0s3
172.25.1.3               ether   08:00:27:26:e6:49   C                     enp0s3
cs2020@ubuntu:~$ python3 Desktop/login.py
cs2020@ubuntu:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.25.1.4               ether   08:00:aa:bb:cc:dd   C                     enp0s3
172.25.1.1               ether   52:54:00:12:35:00   C                     enp0s3
172.25.1.3               ether   08:00:27:26:e6:49   C                     enp0s3
cs2020@ubuntu:~$
```

(3) The change of ARP table in victim side (3 phases of ARP table: before MitM attack, during MitM attack, after MitM attack)

login.py is at victim side, it can input the username and password to http://140.113.207.246/login.php.

login.py:

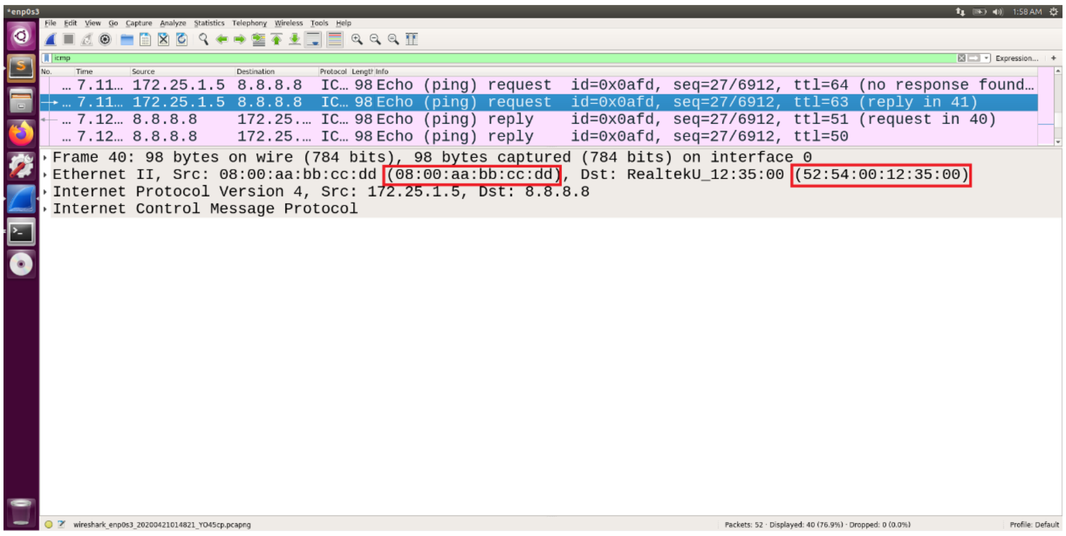
```
import requests
url = "http://140.113.207.246/login.php"
session = requests.Session()
payload = {'usr':'CS2020','pwd':'cs2020','btn_login':'Login'}
r = session.post(url,data = payload)
```


*enp0s3
1:58 AM
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
icmp
Expression…

```
No.  Time    Source      Destination  Protocol Length Info
 …  7.11…  172.25.1.5  8.8.8.8      IC… 98 Echo (ping) request  id=0x0afd, seq=27/6912, ttl=64 (no response found…
 …  7.11…  172.25.1.5  8.8.8.8      IC… 98 Echo (ping) request  id=0x0afd, seq=27/6912, ttl=63 (reply in 41)
 …  7.12…  8.8.8.8     172.25.…     IC… 98 Echo (ping) reply    id=0x0afd, seq=27/6912, ttl=51 (request in 40)
 …  7.12…  8.8.8.8     172.25.…     IC… 98 Echo (ping) reply    id=0x0afd, seq=27/6912, ttl=50
▸ Frame 39: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_1a:e3:a7 (08:00:27:1a:e3:a7), Dst: 08:00:aa:bb:cc:dd (08:00:aa:bb:cc:dd)
▸ Internet Protocol Version 4, Src: 172.25.1.5, Dst: 8.8.8.8
▸ Internet Control Message Protocol
```

```
wireshark_enp0s3_20200421014821_YO45cp.pcapng    Packets: 52 · Displayed: 40 (76.9%) · Dropped: 0 (0.0%)    Profile: Default
```

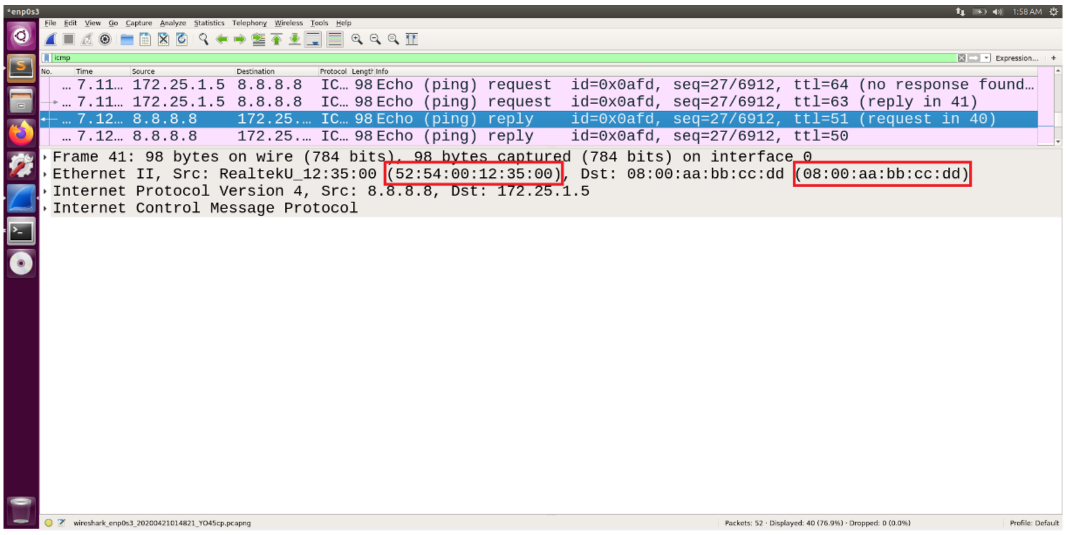
(4-1) Packet sniffing by Wireshark at attacker (Victim => Attacker)


*enp0s3
1:58 AM
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
icmp
Expression…

```
No.  Time    Source      Destination  Protocol Length Info
 …  7.11…  172.25.1.5  8.8.8.8      IC… 98 Echo (ping) request  id=0x0afd, seq=27/6912, ttl=64 (no response found…
 …  7.11…  172.25.1.5  8.8.8.8      IC… 98 Echo (ping) request  id=0x0afd, seq=27/6912, ttl=63 (reply in 41)
 …  7.12…  8.8.8.8     172.25.…     IC… 98 Echo (ping) reply    id=0x0afd, seq=27/6912, ttl=51 (request in 40)
 …  7.12…  8.8.8.8     172.25.…     IC… 98 Echo (ping) reply    id=0x0afd, seq=27/6912, ttl=50
▸ Frame 40: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: 08:00:aa:bb:cc:dd (08:00:aa:bb:cc:dd), Dst: RealtekU_12:35:00 (52:54:00:12:35:00)
▸ Internet Protocol Version 4, Src: 172.25.1.5, Dst: 8.8.8.8
▸ Internet Control Message Protocol
```

```
wireshark_enp0s3_20200421014821_YO45cp.pcapng    Packets: 52 · Displayed: 40 (76.9%) · Dropped: 0 (0.0%)    Profile: Default
```

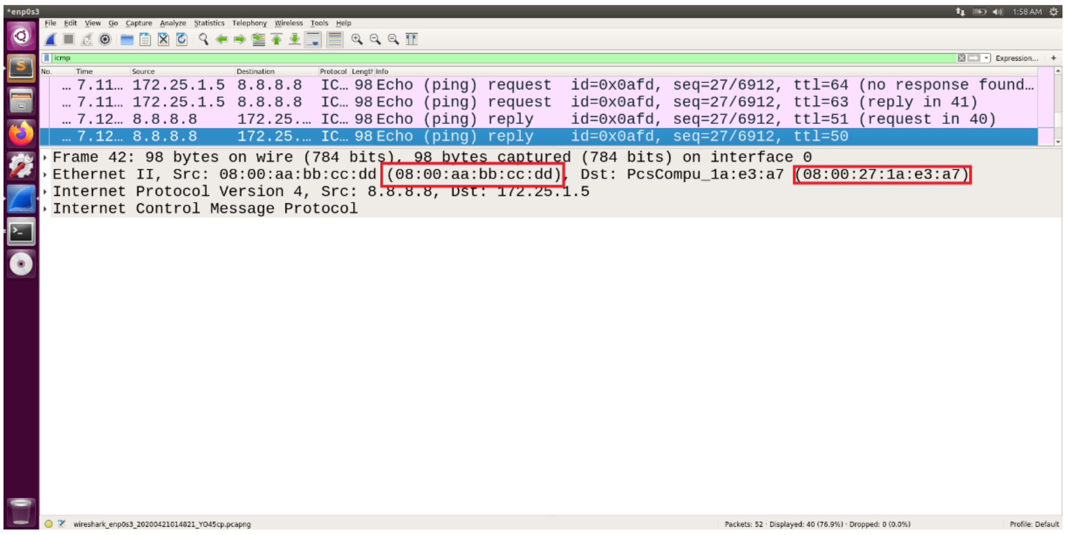
(4-2) Packet sniffing by Wireshark at attacker (Attack => AP)


*enp0s3
1:58 AM
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
icmp
Expression…

```
No.  Time    Source      Destination  Protocol Length Info
 …  7.11…  172.25.1.5  8.8.8.8      IC… 98 Echo (ping) request  id=0x0afd, seq=27/6912, ttl=64 (no response found…
 …  7.11…  172.25.1.5  8.8.8.8      IC… 98 Echo (ping) request  id=0x0afd, seq=27/6912, ttl=63 (reply in 41)
 …  7.12…  8.8.8.8     172.25.…     IC… 98 Echo (ping) reply    id=0x0afd, seq=27/6912, ttl=51 (request in 40)
 …  7.12…  8.8.8.8     172.25.…     IC… 98 Echo (ping) reply    id=0x0afd, seq=27/6912, ttl=50
▸ Frame 41: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: RealtekU_12:35:00 (52:54:00:12:35:00), Dst: 08:00:aa:bb:cc:dd (08:00:aa:bb:cc:dd)
▸ Internet Protocol Version 4, Src: 8.8.8.8, Dst: 172.25.1.5
▸ Internet Control Message Protocol
```

```
wireshark_enp0s3_20200421014821_YO45cp.pcapng    Packets: 52 · Displayed: 40 (76.9%) · Dropped: 0 (0.0%)    Profile: Default
```

(4-3) Packet sniffing by Wireshark at attacker (AP => Attack)



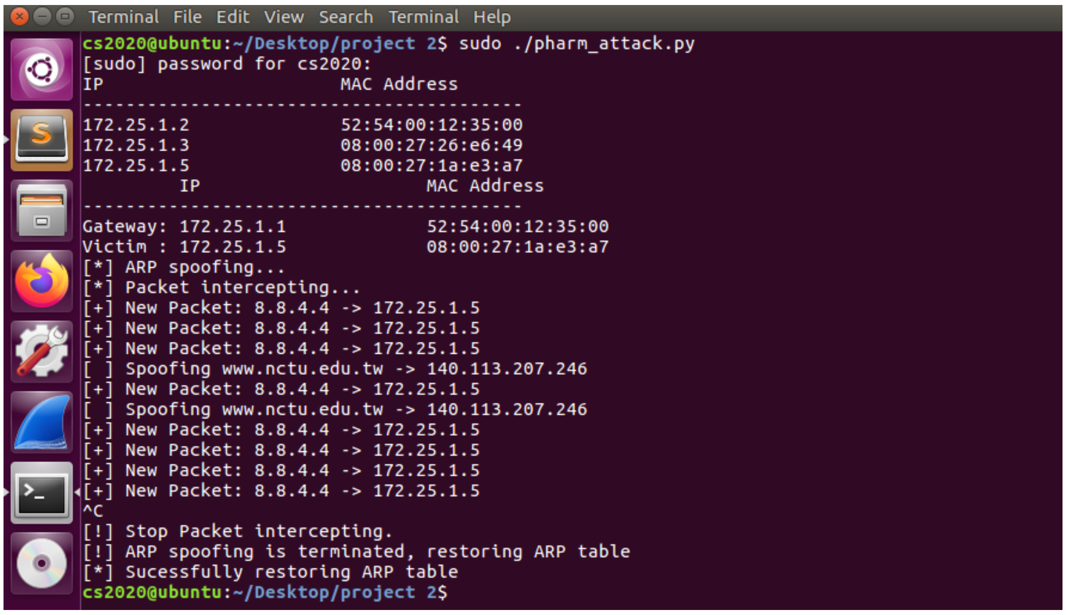
```
No.  Time     Source       Destination  Protocol Length Info
 … 7.11… 172.25.1.5  8.8.8.8    IC… 98 Echo (ping) request  id=0x0afd, seq=27/6912, ttl=64 (no response found…
 … 7.11… 172.25.1.5  8.8.8.8    IC… 98 Echo (ping) request  id=0x0afd, seq=27/6912, ttl=63 (reply in 41)
 … 7.12… 8.8.8.8     172.25.…   IC… 98 Echo (ping) reply    id=0x0afd, seq=27/6912, ttl=51 (request in 40)
 … 7.12… 8.8.8.8     172.25.…   IC… 98 Echo (ping) reply    id=0x0afd, seq=27/6912, ttl=50
▸ Frame 42: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: 08:00:aa:bb:cc:dd (08:00:aa:bb:cc:dd), Dst: PcsCompu_1a:e3:a7 (08:00:27:1a:e3:a7)
▸ Internet Protocol Version 4, Src: 8.8.8.8, Dst: 172.25.1.5
▸ Internet Control Message Protocol
```

(4-4) Packet sniffing by Wireshark at attacker (Attacker => Victim)

By picture (1), (2), (3), (4-1) ~ (4-4), it can show that MitM attack is success.

Part 2: Pharming attack

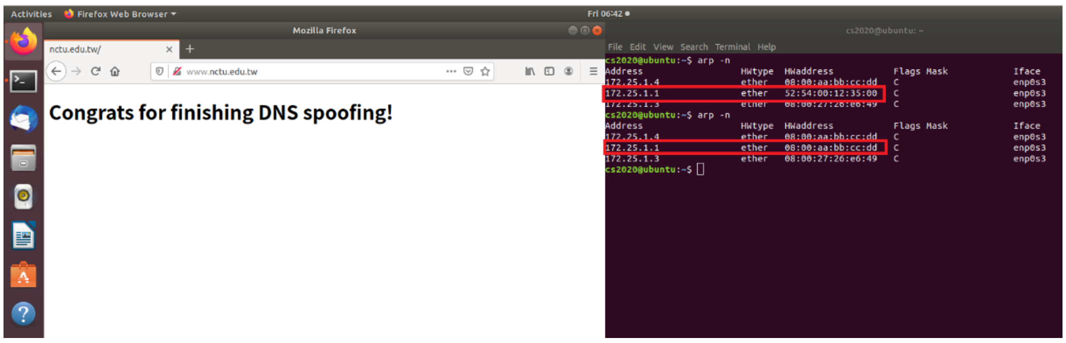
Task III: DNS Spoofing



```
cs2020@ubuntu:~/Desktop/project 2$ sudo ./pharm_attack.py
[sudo] password for cs2020:
IP                     MAC Address
-----------------------------------------
172.25.1.2             52:54:00:12:35:00
172.25.1.3             08:00:27:26:e6:49
172.25.1.5             08:00:27:1a:e3:a7
         IP                     MAC Address
-----------------------------------------
Gateway: 172.25.1.1             52:54:00:12:35:00
Victim : 172.25.1.5             08:00:27:1a:e3:a7
[*] ARP spoofing...
[*] Packet intercepting...
[+] New Packet: 8.8.4.4 -> 172.25.1.5
[+] New Packet: 8.8.4.4 -> 172.25.1.5
[+] New Packet: 8.8.4.4 -> 172.25.1.5
[ ] Spoofing www.nctu.edu.tw -> 140.113.207.246
[+] New Packet: 8.8.4.4 -> 172.25.1.5
[ ] Spoofing www.nctu.edu.tw -> 140.113.207.246
[+] New Packet: 8.8.4.4 -> 172.25.1.5
[+] New Packet: 8.8.4.4 -> 172.25.1.5
[+] New Packet: 8.8.4.4 -> 172.25.1.5
[+] New Packet: 8.8.4.4 -> 172.25.1.5
^C
[!] Stop Packet intercepting.
[!] ARP spoofing is terminated, restoring ARP table
[*] Sucessfully restoring ARP table
cs2020@ubuntu:~/Desktop/project 2$
```

(5) Pharming attack at attacker side

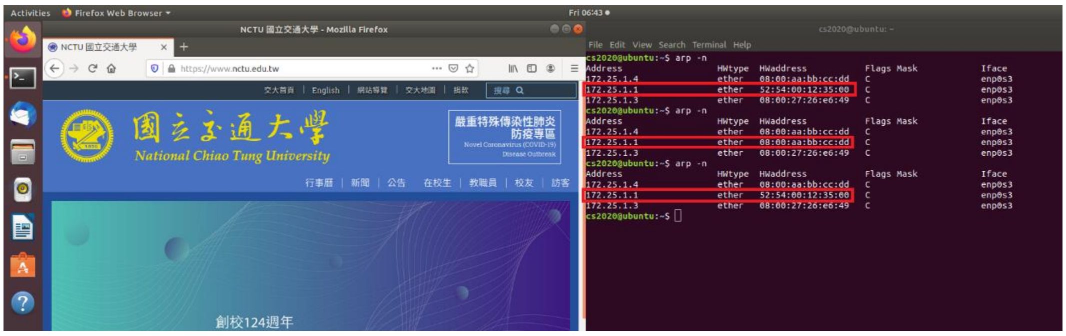
Picture (2) shows the output of ***pharm_attack.py***, including the scanning result, the process and this program will check all DNS response packet from DNS server to victim. Once we notice that the query name in the packet is www.nctu.edu.tw, we will do the DNS spoofing to redirect victim to 140.113.207.246.



Congrats for finishing DNS spoofing!

```
cs2020@ubuntu:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.25.1.4               ether   08:00:aa:bb:cc:dd   C                     enp0s3
172.25.1.1               ether   52:54:00:12:35:00   C                     enp0s3
172.25.1.3               ether   08:00:27:26:e6:49   C                     enp0s3
cs2020@ubuntu:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.25.1.4               ether   08:00:aa:bb:cc:dd   C                     enp0s3
172.25.1.1               ether   08:00:aa:bb:cc:dd   C                     enp0s3
172.25.1.3               ether   08:00:27:26:e6:49   C                     enp0s3
cs2020@ubuntu:~$ 
```

(6-1) ARP table and web page at victim side during pharming attack (2 phases of ARP table: before pharming attack, during pharming attack)



```
cs2020@ubuntu:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.25.1.4               ether   08:00:aa:bb:cc:dd   C                     enp0s3
172.25.1.1               ether   52:54:00:12:35:00   C                     enp0s3
172.25.1.3               ether   08:00:27:26:e6:49   C                     enp0s3
cs2020@ubuntu:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.25.1.4               ether   08:00:aa:bb:cc:dd   C                     enp0s3
172.25.1.1               ether   08:00:aa:bb:cc:dd   C                     enp0s3
172.25.1.3               ether   08:00:27:26:e6:49   C                     enp0s3
cs2020@ubuntu:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.25.1.4               ether   08:00:aa:bb:cc:dd   C                     enp0s3
172.25.1.1               ether   52:54:00:12:35:00   C                     enp0s3
172.25.1.3               ether   08:00:27:26:e6:49   C                     enp0s3
cs2020@ubuntu:~$ 
```

(6-1) ARP table and web page at victim side after pharming attack (3 phases of ARP table: before pharming attack, during pharming attack, after pharming attack)

By picture (5), (6-1), (6-2), it can show that pharming attack is success.

Part 3: Solution to defend against ARP spoofing attack

If the network is small, the ideal way to defend against ARP spoofing is using static ARP entries. DHCP snooping is another way can be considered. Network devices can record the MAC address of all devices in the network by DHCP. Thus, we can detect ARP spoofing attack when receiving an ARP spoofing packet.

Using security protocols that has encryption can also help to reduce the harm of a successful ARP poisoning attack. If victims only use these protocols (ex. HTTPS, SSH) to do data transmit, the attacker can't obtain useful data though attacker has successfully launched ARP spoofing.