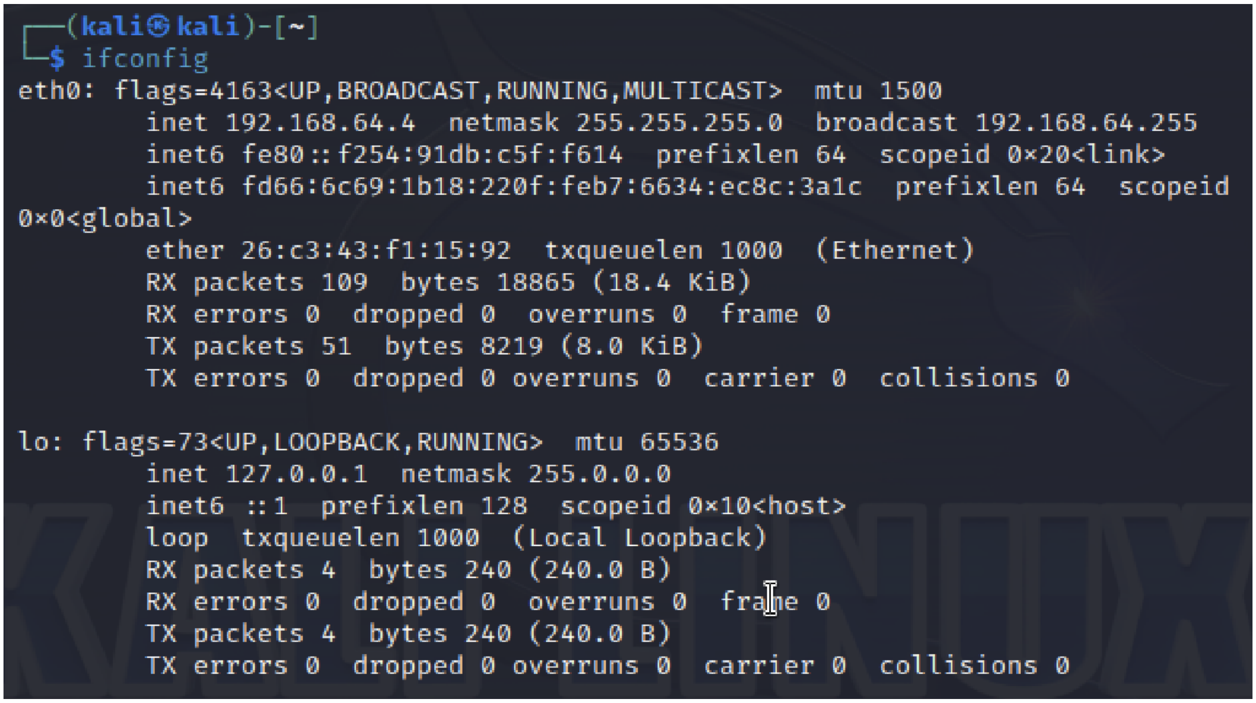
Person-In-The-Middle via Arp Spoofing

Execution

(a) What is Kali’s main interface’s MAC address?

As we can see below (after “ether”) it is **26:c3:43:f1:15:92**.



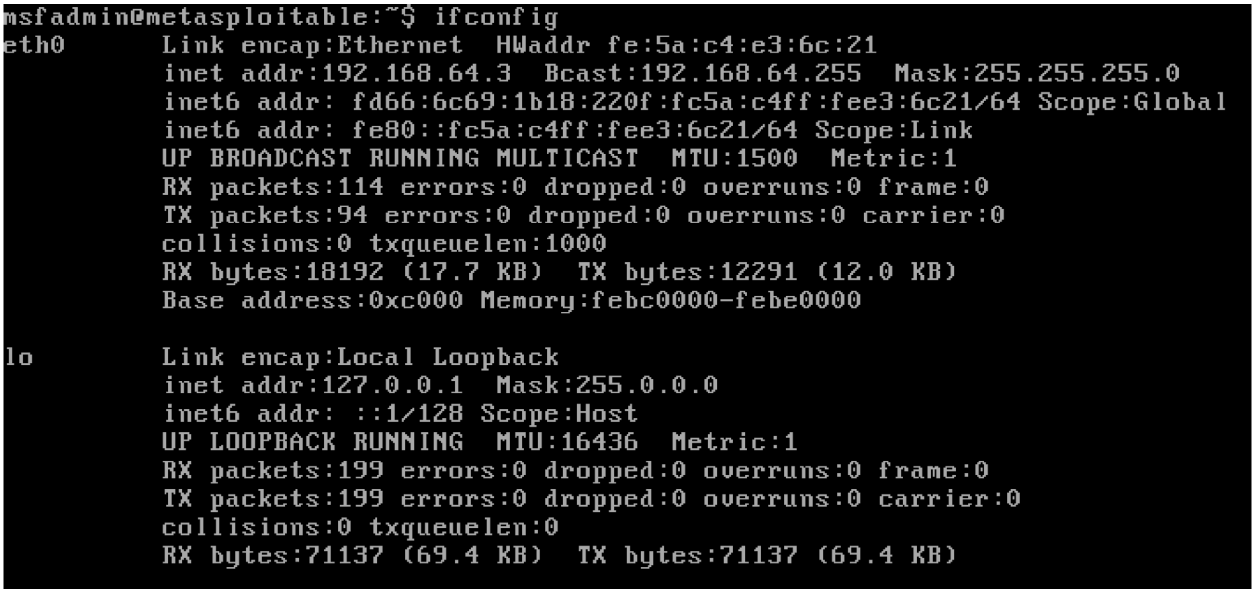
```
┌──(kali㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.64.4  netmask 255.255.255.0  broadcast 192.168.64.255
        inet6 fe80::f254:91db:c5f:f614  prefixlen 64  scopeid 0x20<link>
        inet6 fd66:6c69:1b18:220f:feb7:6634:ec8c:3a1c  prefixlen 64  scopeid
0x0<global>
        ether 26:c3:43:f1:15:92  txqueuelen 1000  (Ethernet)
        RX packets 109  bytes 18865 (18.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 51  bytes 8219 (8.0 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 4  bytes 240 (240.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4  bytes 240 (240.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

(b) What is Kali’s main interface’s IP address?

Kali’s main interface’s IP address is **192.168.64.4**.

(c) What is Metasploitable's main interface's MAC address?

As shown below (after "HWaddr") it is **fe:5a:c4:e3:6c:21**.

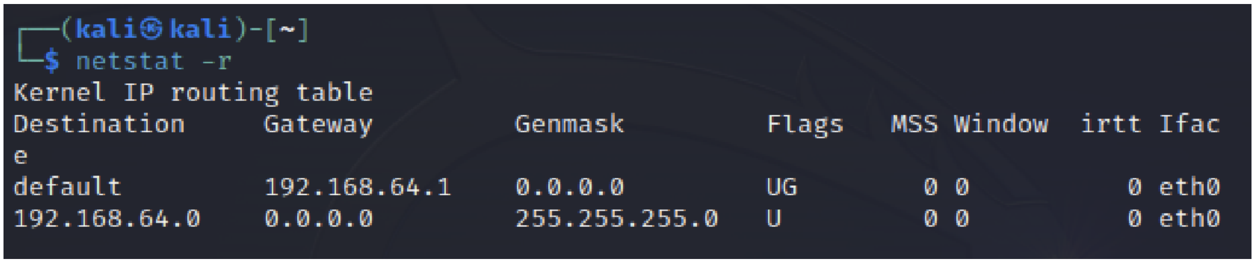


```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr fe:5a:c4:e3:6c:21
          inet addr:192.168.64.3  Bcast:192.168.64.255  Mask:255.255.255.0
          inet6 addr: fd66:6c69:1b18:220f:fc5a:c4ff:fee3:6c21/64 Scope:Global
          inet6 addr: fe80::fc5a:c4ff:fee3:6c21/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:114 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:18192 (17.7 KB)  TX bytes:12291 (12.0 KB)
          Base address:0xc000 Memory:febc0000-febe0000

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:16436  Metric:1
          RX packets:199 errors:0 dropped:0 overruns:0 frame:0
          TX packets:199 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:71137 (69.4 KB)  TX bytes:71137 (69.4 KB)
```

(d) What is Metasploitable's main interface's IP address?

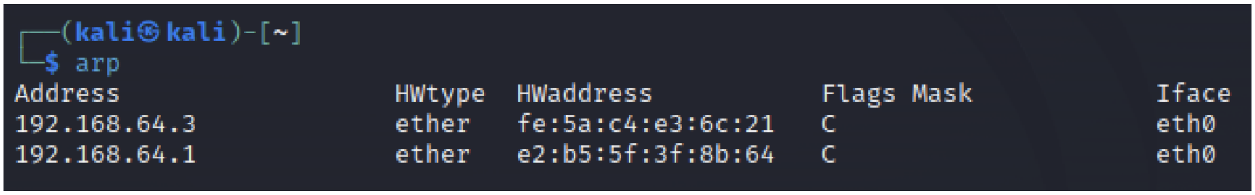
Metasploitable's main interface's IP address is **192.168.64.3**.

(e) Show Kali's routing table.



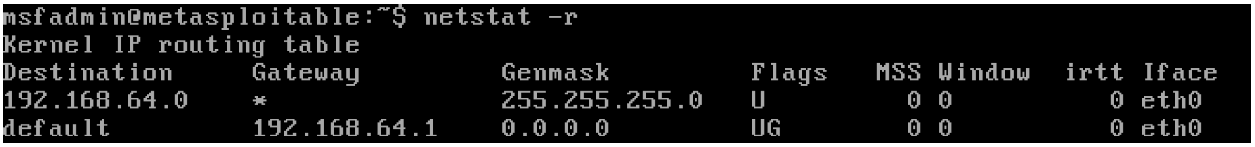
```
┌──(kali㉿kali)-[~]
└─$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Ifac
e
default         192.168.64.1    0.0.0.0         UG        0 0          0 eth0
192.168.64.0    0.0.0.0         255.255.255.0   U         0 0          0 eth0
```

(f) Show Kali's ARP cache.



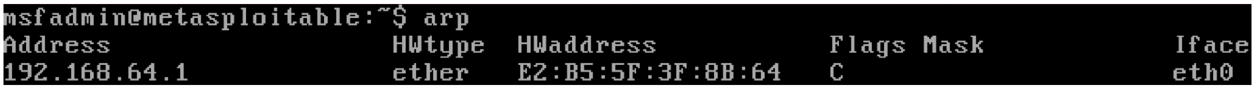
```
┌──(kali㉿kali)-[~]
└─$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.64.3             ether   fe:5a:c4:e3:6c:21   C                     eth0
192.168.64.1             ether   e2:b5:5f:3f:8b:64   C                     eth0
```

(g) Show Metasploitable's routing table.



```
msfadmin@metasploitable:~$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.64.0    *               255.255.255.0   U         0 0          0 eth0
default         192.168.64.1    0.0.0.0         UG        0 0          0 eth0
```

(h) Show Metasploitable's ARP cache.

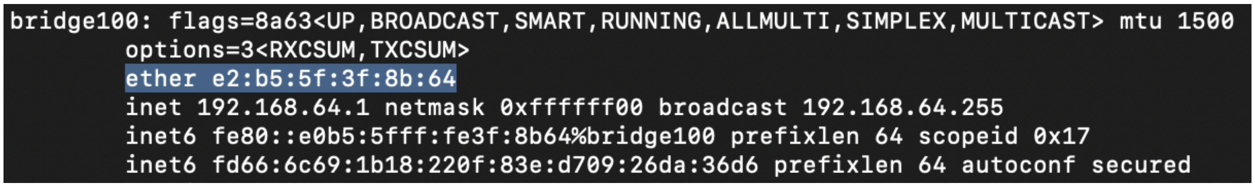


```
msfadmin@metasploitable:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.64.1             ether   E2:B5:5F:3F:8B:64   C                     eth0
```

(i) Suppose the user of Metasploitable wants to get the CS338 sandbox page via the command "curl http://cs338.jeffondich.com/". To which MAC address should Metasploitable send the TCP SYN packet to get the whole HTTP query started? Explain why.

After testing this on Kali with Wireshark, it looks like the TCP SYN packet is sent to the following MAC address: **e2:b5:5f:3f:8b:64**. It's reasonable to assume that the same would happen for Metasploitable, as it is set up symmetrically to Kali on my system.

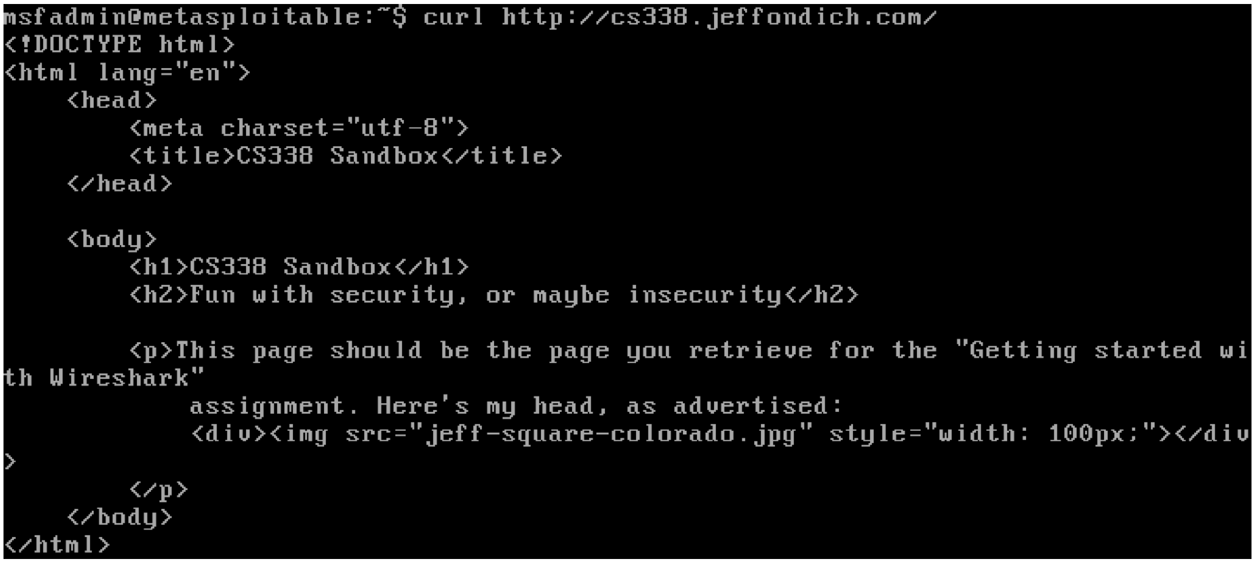
That address is the MAC address of my machine, as shown below. This makes sense since the packet must first hop through my physical machine before reaching its destination.



```
bridge100: flags=8a63<UP,BROADCAST,SMART,RUNNING,ALLMULTI,SIMPLEX,MULTICAST> mtu 1500
	options=3<RXCSUM,TXCSUM>
	ether e2:b5:5f:3f:8b:64
	inet 192.168.64.1 netmask 0xffffff00 broadcast 192.168.64.255
	inet6 fe80::e0b5:5fff:fe3f:8b64%bridge100 prefixlen 64 scopeid 0x17
	inet6 fd66:6c69:1b18:220f:83e:d709:26da:36d6 prefixlen 64 autoconf secured
```

(j) Fire up Wireshark on Kali. Start capturing packets for "tcp port http". On Metasploitable, execute "curl http://cs338.jeffondich.com/". On Kali, stop capturing. Do you see an HTTP response on Metasploitable? Do you see any captured packets in Wireshark on Kali?

After running this command on Metasploitable, I got an HTTP response on Metasploitable:



```
msfadmin@metasploitable:~$ curl http://cs338.jeffondich.com/
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>CS338 Sandbox</title>
    </head>

    <body>
        <h1>CS338 Sandbox</h1>
        <h2>Fun with security, or maybe insecurity</h2>

        <p>This page should be the page you retrieve for the "Getting started wi
th Wireshark"
            assignment. Here's my head, as advertised:
            <div><img src="jeff-square-colorado.jpg" style="width: 100px;"></div
>
        </p>
    </body>
</html>
```

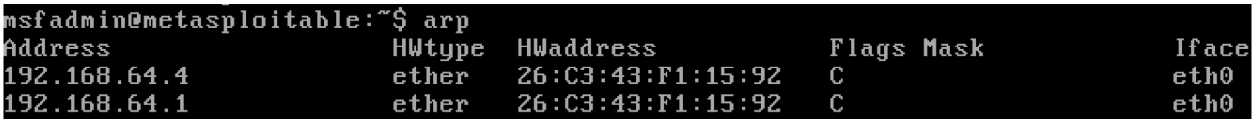
However, Kali did not capture any packets, as packets did not originate from or arrive at Kali.

(k) Do ARP Poisoning

Got this set up and working!

(l) Show Metasploitable's ARP cache. How has it changed?

Here is Metasploitable's new ARP cache.



```
msfadmin@metasploitable:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.64.4             ether   26:C3:43:F1:15:92   C                     eth0
192.168.64.1             ether   26:C3:43:F1:15:92   C                     eth0
```

One change is that we now have Kali added to the ARP cache. More importantly, both my MacBook and Kali now have the MAC address of Kali associated with them on Metasploitable (**26:c3:43:f1:15:92**).

(m) Without actually doing it yet, predict what will happen if you execute "curl http://cs338.jeffondich.com/" on Metasploitable now. Specifically, to what MAC address will Metasploitable send the TCP SYN packet? Explain why.

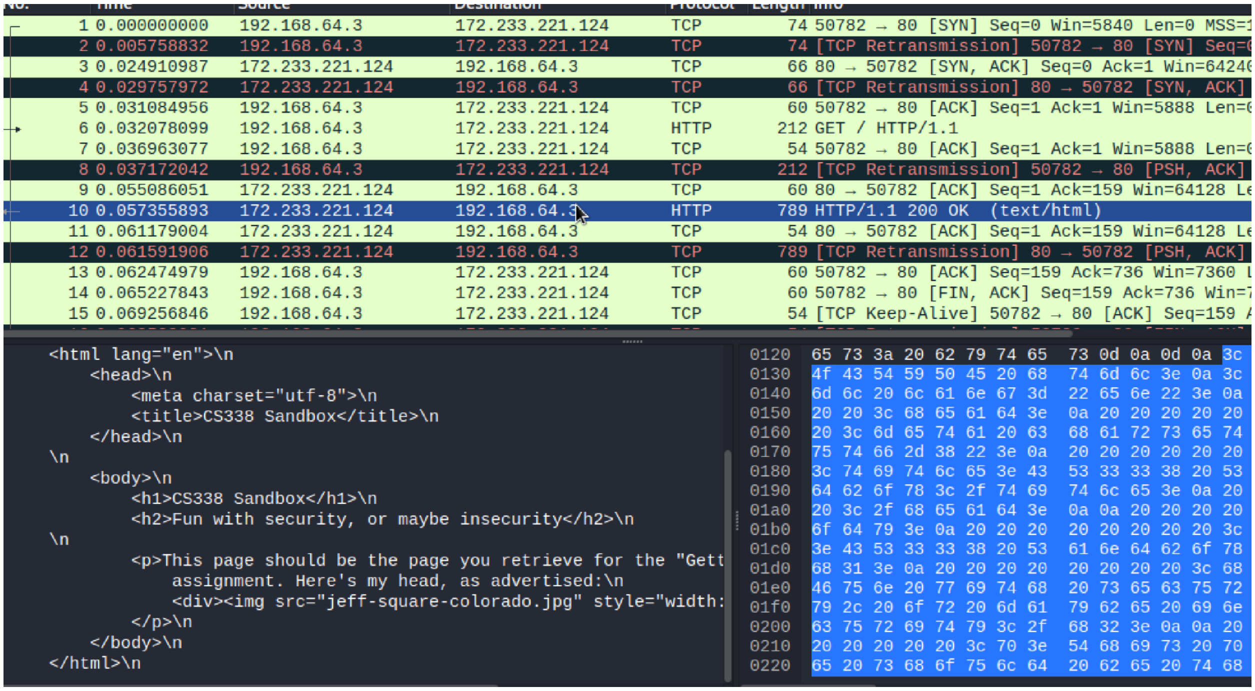
I guess that Metasploitable will send the TCP SYN packet to Kali's MAC address, as this is now the MAC address associated with my MacBook in Metasploitable's ARP cache. The packet still needs to hop through my machine, so it will need to be addressed to “my machine's” MAC address.

(n) Start Wireshark capturing "tcp port http" again.

Done.

(o) Execute "curl http://cs338.jeffondich.com/" on Metasploitable. On Kali, stop capturing. Do you see an HTTP response on Metasploitable? Do you see captured packets in Wireshark? Can you tell from Kali what messages went back and forth between Metasploitable and cs338.jeffondich.com?

Again, I do see an HTTP response on Metaspoitable. However, this time, I did capture packets in Wireshark. I can see every message that went back and forth between Metasploitable and cs338.jeffondich.com. Here’s a snapshot showing the HTTP response that Metasploitable received (on Kali).



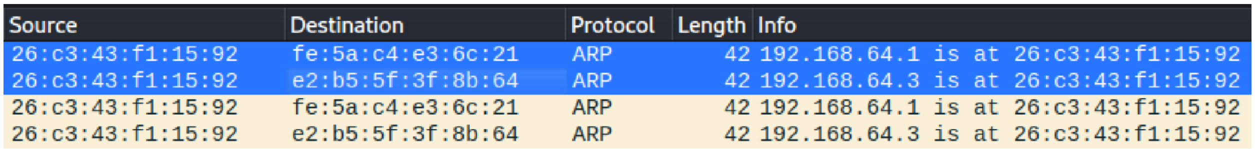
```
No.  Time        Source           Destination      Protocol Length Info
   1 0.000000000 192.168.64.3     172.233.221.124  TCP       74 50782 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1
   2 0.005758832 192.168.64.3     172.233.221.124  TCP       74 [TCP Retransmission] 50782 → 80 [SYN] Seq=0
   3 0.024910987 172.233.221.124  192.168.64.3     TCP       66 80 → 50782 [SYN, ACK] Seq=0 Ack=1 Win=64240
   4 0.029757972 172.233.221.124  192.168.64.3     TCP       66 [TCP Retransmission] 80 → 50782 [SYN, ACK]
   5 0.031084956 192.168.64.3     172.233.221.124  TCP       60 50782 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0
   6 0.032078099 192.168.64.3     172.233.221.124  HTTP     212 GET / HTTP/1.1
   7 0.036963077 192.168.64.3     172.233.221.124  TCP       54 50782 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0
   8 0.037172042 192.168.64.3     172.233.221.124  TCP      212 [TCP Retransmission] 50782 → 80 [PSH, ACK]
   9 0.055086051 172.233.221.124  192.168.64.3     TCP       60 80 → 50782 [ACK] Seq=1 Ack=159 Win=64128 Le
  10 0.057355893 172.233.221.124  192.168.64.3     HTTP     789 HTTP/1.1 200 OK  (text/html)
  11 0.061179004 172.233.221.124  192.168.64.3     TCP       54 80 → 50782 [ACK] Seq=1 Ack=159 Win=64128 Le
  12 0.061591906 172.233.221.124  192.168.64.3     TCP      789 [TCP Retransmission] 80 → 50782 [PSH, ACK]
  13 0.062474979 192.168.64.3     172.233.221.124  TCP       60 50782 → 80 [ACK] Seq=159 Ack=736 Win=7360 L
  14 0.065227843 192.168.64.3     172.233.221.124  TCP       60 50782 → 80 [FIN, ACK] Seq=159 Ack=736 Win=7
  15 0.069256846 192.168.64.3     172.233.221.124  TCP       54 [TCP Keep-Alive] 50782 → 80 [ACK] Seq=159 A
```

```
<html lang="en">\n
    <head>\n
        <meta charset="utf-8">\n
        <title>CS338 Sandbox</title>\n
    </head>\n
\n
    <body>\n
        <h1>CS338 Sandbox</h1>\n
        <h2>Fun with security, or maybe insecurity</h2>\n
\n
        <p>This page should be the page you retrieve for the "Gett
            assignment. Here's my head, as advertised:\n
            <div><img src="jeff-square-colorado.jpg" style="width:
        </p>\n
    </body>\n
</html>\n
```

```
0120  65 73 3a 20 62 79 74 65  73 0d 0a 0d 0a 3c
0130  4f 43 54 59 50 45 20 68  74 6d 6c 3e 0a 3c
0140  6d 6c 20 6c 61 6e 67 3d  22 65 6e 22 3e 0a
0150  20 20 3c 68 65 61 64 3e  0a 20 20 20 20 20
0160  20 3c 6d 65 74 61 20 63  68 61 72 73 65 74
0170  75 74 66 2d 38 22 3e 0a  20 20 20 20 20 20
0180  3c 74 69 74 6c 65 3e 43  53 33 33 38 20 53
0190  64 62 6f 78 3c 2f 74 69  74 6c 65 3e 0a 20
01a0  20 3c 2f 68 65 61 64 3e  0a 0a 20 20 20 20
01b0  6f 64 79 3e 0a 20 20 20  20 20 20 20 20 3c
01c0  3e 43 53 33 33 38 20 53  61 6e 64 62 6f 78
01d0  68 31 3e 0a 20 20 20 20  20 20 20 20 3c 68
01e0  46 75 6e 20 77 69 74 68  20 73 65 63 75 72
01f0  79 2c 20 6f 72 20 6d 61  79 62 65 20 69 6e
0200  63 75 72 69 74 79 3c 2f  68 32 3e 0a 0a 20
0210  20 20 20 20 20 3c 70 3e  54 68 69 73 20 70
0220  65 20 73 68 6f 75 6c 64  20 62 65 20 74 68
```

(p) Explain in detail what happened. How did Kali change Metasploitable’s ARP cache?

We can watch the attack in action by capturing ARP packets:



Source	Destination	Protocol	Length	Info
26:c3:43:f1:15:92	fe:5a:c4:e3:6c:21	ARP	42	192.168.64.1 is at 26:c3:43:f1:15:92
26:c3:43:f1:15:92	e2:b5:5f:3f:8b:64	ARP	42	192.168.64.3 is at 26:c3:43:f1:15:92
26:c3:43:f1:15:92	fe:5a:c4:e3:6c:21	ARP	42	192.168.64.1 is at 26:c3:43:f1:15:92
26:c3:43:f1:15:92	e2:b5:5f:3f:8b:64	ARP	42	192.168.64.3 is at 26:c3:43:f1:15:92

As we can see, two ARP announcement packets are sent repeatedly:

- One is sent to my MacBook, saying “The MAC address for Metasploitable is [Kali’s MAC address”
- The other is sent to Metasploitable, saying “The MAC address for Varun’s MacBook is [Kali’s MAC address]”

This way, for any communications passing between Metasploitable and my MacBook, Kali can act as the AITM after these MAC addresses are cached. I believe that these packets are sent repeatedly because ARP caches are periodically cleared.

(q) If you wanted to design an ARP spoofing detector, what would you have your detector do?

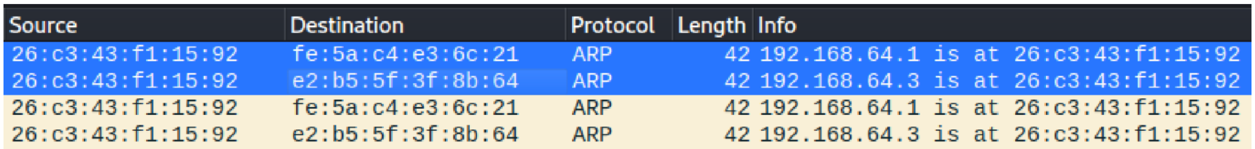
Something that I noticed about the ARP poisoning attack was that it requires several ARP packets to be sent since the cache of each machine gets cleared over time. A protective measure would be to flag frequent ARP messages as suspicious. This is susceptible to false positives, however, as even “normal” ARP packets may need to be sent frequently.

A heavy-duty solution could be to hardcode MAC addresses for devices on the same network. Perhaps only a single trusted device could broadcast MAC/IP address updates (like the network’s router), using encryption/a digital signature that identifies the source as valid. Then, any ARP packet that conflicts with hardcoded or “trusted” information would be detected as suspicious.

Synthesis

(a) Explain in detail Mal’s strategy for intercepting the traffic between Alice and Bob. Use any of your observations from the Execution section to clarify your explanation.

The following picture (also shown above) is fairly illustrative of how Mal acts as an AITM.



Source	Destination	Protocol	Length	Info
26:c3:43:f1:15:92	fe:5a:c4:e3:6c:21	ARP	42	192.168.64.1 is at 26:c3:43:f1:15:92
26:c3:43:f1:15:92	e2:b5:5f:3f:8b:64	ARP	42	192.168.64.3 is at 26:c3:43:f1:15:92
26:c3:43:f1:15:92	fe:5a:c4:e3:6c:21	ARP	42	192.168.64.1 is at 26:c3:43:f1:15:92
26:c3:43:f1:15:92	e2:b5:5f:3f:8b:64	ARP	42	192.168.64.3 is at 26:c3:43:f1:15:92

Essentially, Mal can intercept traffic between Alice and Bob by placing herself between Alice and Alice’s first hop (on the way to Bob). Mal can accomplish this by sending out two ARP announcement packets (repeatedly):

- One to Alice saying that Alice’s first hop’s MAC address is [Mal’s MAC address]
- One to Alice’s first hop saying that Alice’s MAC address is [Mal’s MAC address]

After Alice and Alice’s first hop update their ARP tables to reflect this, Mal can intercept packets sent between the two devices. In other words, communication between Alice and Alice’s first hop always passes through Mal, regardless of the direction in which data is traveling.

(b) From Alice’s perspective, is this attack detectable? If not, why not? If so, how would Alice’s setup need to change to detect the attack?

Probably not: if Alice’s ARP table were constantly being cleared and she had no other memory of MAC/IP relationships, then a new suspicious MAC/IP pairing being broadcasted by Mal would be undetectable. To make this attack detectable, Alice would need some form of persistent storage for remembering past MAC/IP pairs to flag newly broadcasted MAC/IP pairs that conflict with older information.

(c) From Bob’s perspective, is this attack detectable?

No: Bob is unable to see anything past his first hop (in the direction of Alice). In other words, Mal is protected by all the hops that Bob’s packets need to make to get to Alice.

(d) Could Alice or Bob detect and/or prevent this attack if the website in question was using HTTPS instead of HTTP? Explain.

I believe that using HTTPS could provide some protection, as Mal wouldn’t be able to decrypt any of the information being shared between Alice and Bob. However, there are still issues that

HTTPS doesn’t resolve. For example, Mal could still perform a replay attack by sending Bob copies of Alice’s packets. This could potentially cause damage (e.g. imagine Mal adding forum posts in a tight loop to Bob’s database and blaming Alice for the damages).