Author: Chi Nguyen

a. Kali's main interface's MAC address is 00:0c:29:8a:81:9b.
b. Kali's main interface's IP address is 192.168.63.128.
c. Metasploitable's main interface's MAC address is 00:0c:29:20:2d:34.
d. Metasploitable's main interface's IP address is 192.168.63.129.
e. Kali's routing table:

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.63.2    0.0.0.0         UG        0 0          0 eth0
192.168.63.0    0.0.0.0         255.255.255.0   U         0 0          0 eth0
```

f. Kali's ARP cache:

```
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.63.1             ether   f6:34:f0:4e:2b:65   C                     eth0
192.168.63.254           ether   00:50:56:eb:d0:fa   C                     eth0
192.168.63.129           ether   00:0c:29:20:2d:34   C                     eth0
192.168.63.2             ether   00:50:56:f9:ef:93   C                     eth0
```

g. Metasploitable's routing table:

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.63.0    *               255.255.255.0   U         0 0          0 eth0
default         192.168.63.2    0.0.0.0         UG        0 0          0 eth0
```

h. Metasploitable's ARP cache:

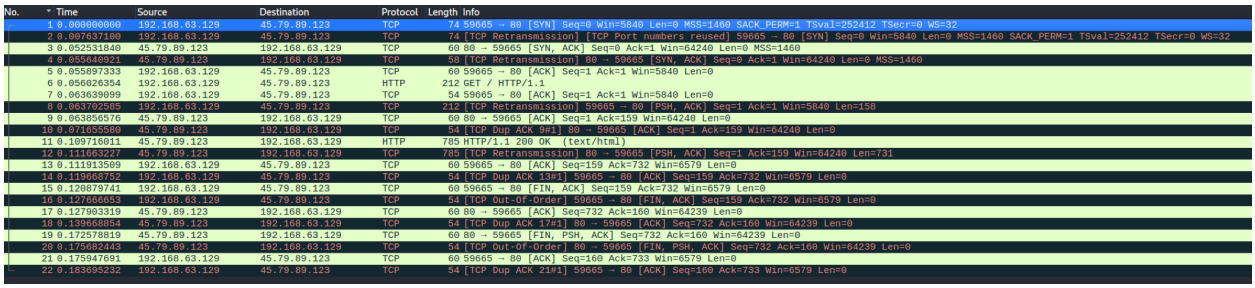
```
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.63.2             ether   00:50:56:F9:EF:93   C                     eth0
192.168.63.128           ether   00:0C:29:8A:81:9B   C                     eth0
```

i. If the user of Metasploitable wants to get the CS338 sandbox page via the command "curl http://cs338.jeffondich.com/", Metasploitable should send the TCP SYN packet to the Metasploitable's MAC address because we are trying to set up a connection between the sandbox server and the Metasploitable machine itself.
j. There is HTTP response on Metasploitable but I don't see any captured packets in Wireshark on Kali.
k. Doing stuff…
l. Metasploitable's ARP cache during the poisoning:

```
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.63.2             ether   00:0C:29:8A:81:9B   C                     eth0
192.168.63.128           ether   00:0C:29:8A:81:9B   C                     eth0
192.168.63.254           ether   00:0C:29:8A:81:9B   C                     eth0
192.168.63.1             ether   00:0C:29:8A:81:9B   C                     eth0
```

Here, we can see that there are two new addresses (192.168.63.1 and 192.168.63.254). This table looks like Kali's ARP cache before the poisoning.
m. If I execute "curl http://cs338.jeffondich.com/" on Metasploitable now, the MAC address that Metasploitable will send the TCP SYN packet to will be Kali's MAC address. This is because we are doing ARP poisoning on the Kali machine, so the connection will be set up so that Metasploitable send the packet to Kali.
n. Doing it…

o. There is an HTTP response on Metasploitable, and there are 22 captured packets in Wireshark, and it is entirely possible to tell from Kali which messages went back and forth between Metasploitable and cs338.jeffondich.com. As shown in the screenshot attached below, we can see all the HTTP headers with the GET request and the 200 OK responses and what actually are exchanged within this connection.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.63.129	45.79.89.123	TCP	74	59665 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=252412 TSecr=0 WS=32
2	0.007637100	192.168.63.129	45.79.89.123	TCP	74	[TCP Retransmission] [TCP Port numbers reused] 59665 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=252412 TSecr=0 WS=32
3	0.052531840	45.79.89.123	192.168.63.129	TCP	60	80 → 59665 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
4	0.055640921	45.79.89.123	192.168.63.129	TCP	58	[TCP Retransmission] 80 → 59665 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
5	0.055897333	192.168.63.129	45.79.89.123	TCP	60	59665 → 80 [ACK] Seq=1 Ack=1 Win=5840 Len=0
6	0.056026354	192.168.63.129	45.79.89.123	HTTP	212	GET / HTTP/1.1
7	0.063639099	192.168.63.129	45.79.89.123	TCP	54	59665 → 80 [ACK] Seq=1 Ack=1 Win=5840 Len=0
8	0.063702583	192.168.63.129	45.79.89.123	TCP	212	[TCP Retransmission] 59665 → 80 [PSH, ACK] Seq=1 Ack=1 Win=5840 Len=158
9	0.063856976	45.79.89.123	192.168.63.129	TCP	60	80 → 59665 [ACK] Seq=1 Ack=159 Win=64240 Len=0
10	0.071055580	45.79.89.123	192.168.63.129	TCP	54	[TCP Dup ACK 9#1] 80 → 59665 [ACK] Seq=1 Ack=159 Win=64240 Len=0
11	0.109716011	45.79.89.123	192.168.63.129	HTTP	785	HTTP/1.1 200 OK (text/html)
12	0.111663227	45.79.89.123	192.168.63.129	TCP	785	[TCP Retransmission] 80 → 59665 [PSH, ACK] Seq=1 Ack=159 Win=64240 Len=731
13	0.111913509	192.168.63.129	45.79.89.123	TCP	60	59665 → 80 [ACK] Seq=159 Ack=732 Win=6579 Len=0
14	0.119068752	192.168.63.129	45.79.89.123	TCP	54	[TCP Dup ACK 13#1] 59665 → 80 [ACK] Seq=159 Ack=732 Win=6579 Len=0
15	0.120879741	192.168.63.129	45.79.89.123	TCP	60	59665 → 80 [FIN, ACK] Seq=159 Ack=732 Win=6579 Len=0
16	0.127666653	192.168.63.129	45.79.89.123	TCP	54	[TCP Out-Of-Order] 59665 → 80 [FIN, ACK] Seq=159 Ack=732 Win=6579 Len=0
17	0.127903319	45.79.89.123	192.168.63.129	TCP	60	80 → 59665 [ACK] Seq=732 Ack=160 Win=64239 Len=0
18	0.139668854	45.79.89.123	192.168.63.129	TCP	54	[TCP Dup ACK 17#1] 80 → 59665 [ACK] Seq=732 Ack=160 Win=64239 Len=0
19	0.172578819	45.79.89.123	192.168.63.129	TCP	60	80 → 59665 [FIN, PSH, ACK] Seq=732 Ack=160 Win=64239 Len=0
20	0.175082443	45.79.89.123	192.168.63.129	TCP	54	[TCP Out-Of-Order] 80 → 59665 [FIN, PSH, ACK] Seq=732 Ack=160 Win=64239 Len=0
21	0.175947691	192.168.63.129	45.79.89.123	TCP	60	59665 → 80 [ACK] Seq=160 Ack=733 Win=6579 Len=0
22	0.183695232	192.168.63.129	45.79.89.123	TCP	54	[TCP Dup ACK 21#1] 59665 → 80 [ACK] Seq=160 Ack=733 Win=6579 Len=0

p. Whenever the client or a server send a packet, it seems like ARP entries are duplicated and we have the router troubleshooting in the black-red frames.