

## PERSON-IN-THE-MIDDLE VIA ARP SPOOFING

- What is Kali's main interface's MAC address? (The main interface is probably called eth0, but check ifconfig to be sure.)  
00:0c:29:2f:90:c6
- What is Kali's main interface's IP address?  
192.168.221.129
- What is Metasploitable's main interface's MAC address?  
00:0c:29:35:f7:36
- What is Metasploitable's main interface's IP address?  
192.168.221.128
- Show Kali's routing table. (Use "netstat -r" to see it with symbolic names, or "netstat -rn" to see it with numerical addresses.)

```
(kali㉿kali)-[~]  
$ netstat -r  
Kernel IP routing table  
Destination      Gateway          Genmask          Flags   MSS Window  irtt Iface  
default          192.168.221.2    0.0.0.0          UG      0 0        0 eth0  
192.168.221.0    0.0.0.0          255.255.255.0    U        0 0        0 eth0
```

- Show Kali's ARP cache. (Use "arp" or "arp -n".)

```
(kali㉿kali)-[~]  
$ arp -n  
Address          HWtype  HWaddress         Flags Mask          Iface  
192.168.221.254  ether   00:50:56:f4:54:d6 C                eth0  
192.168.221.128  ether   00:0c:29:35:f7:36 C                eth0  
192.168.221.2    ether   00:50:56:e8:1d:31 C                eth0
```

- Show Metasploitable's routing table.

```
msfadmin@metasploitable:~$ netstat -r  
Kernel IP routing table  
Destination      Gateway          Genmask          Flags   MSS Window  irtt Iface  
192.168.221.0    *               255.255.255.0    U        0 0        0 eth0  
default          192.168.221.2    0.0.0.0          UG      0 0        0 eth0  
msfadmin@metasploitable:~$
```

- Show Metasploitable's ARP cache.

```
msfadmin@metasploitable:~$ arp -n  
Address          HWtype  HWaddress         Flags Mask          Iface  
192.168.221.2    ether   00:50:56:E8:1D:31 C                eth0  
192.168.221.129  ether   00:0C:29:2F:90:C6 C                eth0  
msfadmin@metasploitable:~$
```

- 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.

Metasploitable uses the MAC address in the ARP table, which is 00:0c:29:35:f7:36

- 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?

We see the HTTP response on Metasploitable but we don't see any captured packets in wireshark.

- k) Now, it's time to be Mal (who will, today, merely eavesdrop). Use Ettercap to do ARP spoofing (also known as ARP Cache Poisoning) with Metasploitable as your target. There are many online tutorials on how to do this (here's one). Find one you like, and start spoofing your target. NOTE: most of these tutorials are showing an old user interface for Ettercap, which may make them confusing. The steps you're trying to take within Ettercap are:

- i) Start sniffing (not bridged sniffing) on eth0
- ii) Scan for Hosts
- iii) View the Hosts list
- iv) Select your Metasploit VM from the Host List
- v) Add that host as Target 1
- vi) Start ARP Poisoning (including Sniff Remote Connections)
- vii) Do your stuff with wireshark and Metasploit
- viii) Stop ARP Poisoning

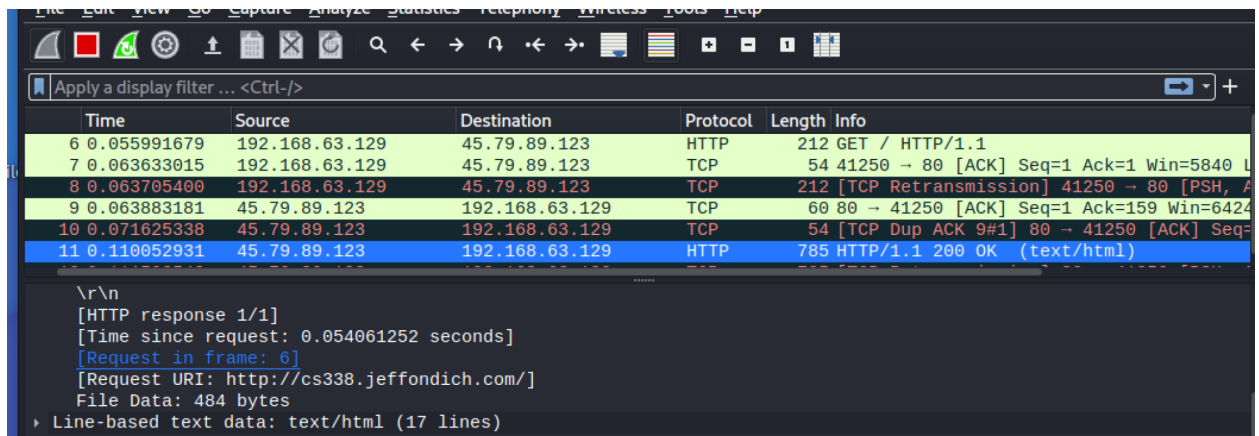
I'll post some screenshots on Slack of how I got Ettercap to do these things. Honestly, I don't know who redesigned this user interface to make it so much harder to do things, but they did. (Common enough in the Linux UI world.)

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

```
msfadmin@metasploitable:~$ arp -n
Address      HWtype  HWaddress    Flags Mask    Iface
192.168.63.1 ether    F6:34:F0:4E:2B:65 C             eth0
192.168.63.2 ether    00:50:56:F9:EF:93 C             eth0
192.168.63.254 ether    00:50:56:F9:89:F1 C             eth0
msfadmin@metasploitable:~$ arp -n
Address      HWtype  HWaddress    Flags Mask    Iface
192.168.63.1 ether    00:0C:29:8A:81:9B C             eth0
192.168.63.2 ether    00:0C:29:8A:81:9B C             eth0
192.168.63.254 ether    00:0C:29:8A:81:9B C             eth0
msfadmin@metasploitable:~$
```

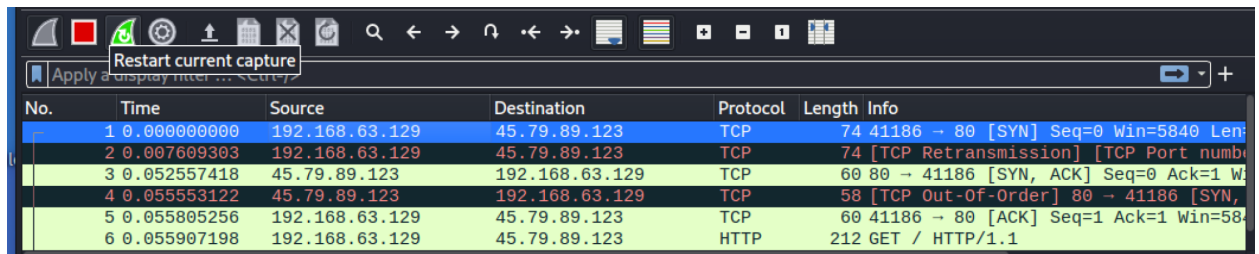
Now, all the MAC addresses have been changed by the spoofing. They are all the same now.

- 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.

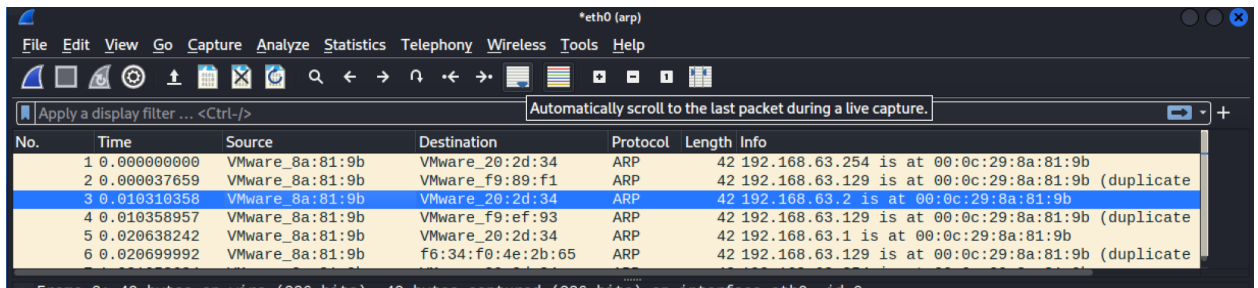
It will send the request to the wrong address (00:0c:29:8a:81:9b) since the arp table has been changed. Metasploitable has no way of knowing that the address it was given was wrong.



- n) Start Wireshark capturing "tcp port http" again.
- 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?  
 Yes. We are able to see the SYN ACK packet as well as the HTTP 200 OK response and everything else. We have successfully done a MITM attack.
- p) Explain in detail what happened. How did Kali change Metasploitable's ARP cache? (If you want to watch the attack in action, try stopping the PITM/MITM attack by selecting "Stop mitm attack(s)" from Ettercap's Mitm menu, starting a Wireshark capture for "arp", and restarting the ARP poisoning attack in Ettercap.)

Kali broadcasted repeated ARP messages to Metasploitable that told what IP address went with what (wrong) MAC addresses. Thus, Metasploitable updated its ARP table

with the incorrect info.



The image shows a Wireshark packet capture window titled '\*eth0 (arp)'. The interface includes a menu bar (File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, Help) and a toolbar. A display filter is set to 'Apply a display filter ... <Ctrl-F>'. A status bar at the top right indicates 'Automatically scroll to the last packet during a live capture.' The packet list table shows six ARP requests, all from source 'VMware\_8a:81:9b'. Packets 2, 4, and 6 are highlighted in blue. Packet 2 is a duplicate of packet 1. Packets 4 and 5 are also duplicates of packet 1. The 'Info' column for each packet shows details like '42 192.168.63.254 is at 00:0c:29:8a:81:9b'.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	VMware_8a:81:9b	VMware_20:2d:34	ARP	42	192.168.63.254 is at 00:0c:29:8a:81:9b
2	0.000037659	VMware_8a:81:9b	VMware_f9:89:f1	ARP	42	192.168.63.129 is at 00:0c:29:8a:81:9b (duplicate)
3	0.010310358	VMware_8a:81:9b	VMware_20:2d:34	ARP	42	192.168.63.2 is at 00:0c:29:8a:81:9b
4	0.010358957	VMware_8a:81:9b	VMware_f9:ef:93	ARP	42	192.168.63.129 is at 00:0c:29:8a:81:9b (duplicate)
5	0.020638242	VMware_8a:81:9b	VMware_20:2d:34	ARP	42	192.168.63.1 is at 00:0c:29:8a:81:9b
6	0.020699992	VMware_8a:81:9b	f6:34:f0:4e:2b:65	ARP	42	192.168.63.129 is at 00:0c:29:8a:81:9b (duplicate)

- q) If you wanted to design an ARP spoofing detector, what would you have your detector do? (As you think about this, consider under what circumstances your detector might generate false positives.)

Ideally, we'd want there to be no duplicates in IP or in Mac address. However, if someone's IP address changed by using a VPN or something, that may cause a false positive. I think blocking/flagging packets for suspicious IP addresses would also work or flagging the repeated and consistent sending of unrequested ARP packets.