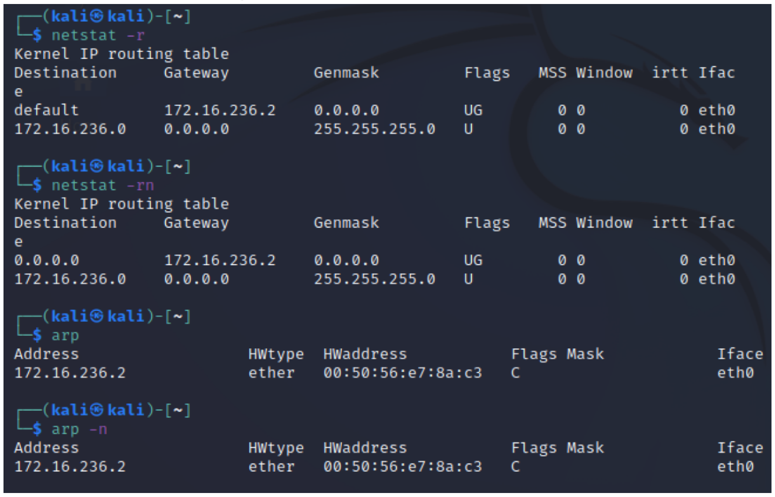
Lane Maitland

- Kali main interface MAC address: 00:0c:29:b3:59:dd
- Kali main interface IP address: 172.16.236.128
- Metasploitable main interface MAC address: 00:0c:29:46:e5:0f
- Metasploitable main interface IP address: 172.16.236.129
- Kali routing table and ARP cache:



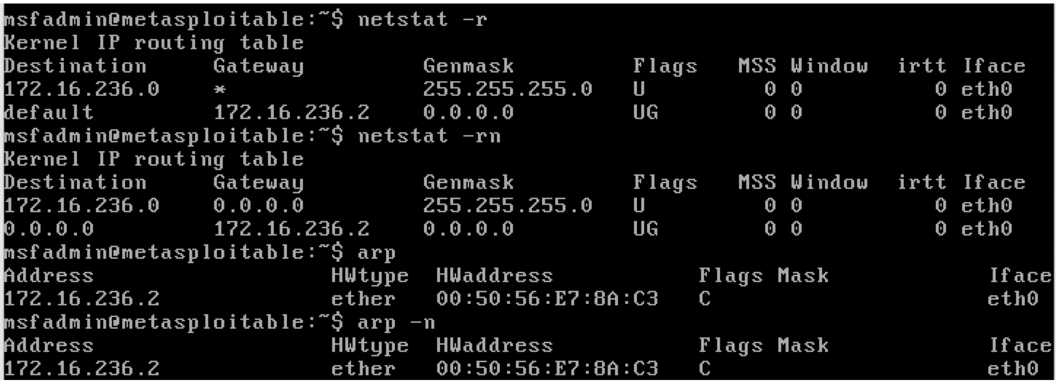
```
┌──(kali㉿kali)-[~]
└─$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Ifac
e
default         172.16.236.2    0.0.0.0         UG        0 0          0 eth0
172.16.236.0    0.0.0.0         255.255.255.0   U         0 0          0 eth0

┌──(kali㉿kali)-[~]
└─$ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Ifac
e
0.0.0.0         172.16.236.2    0.0.0.0         UG        0 0          0 eth0
172.16.236.0    0.0.0.0         255.255.255.0   U         0 0          0 eth0

┌──(kali㉿kali)-[~]
└─$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
172.16.236.2             ether   00:50:56:e7:8a:c3   C                     eth0

┌──(kali㉿kali)-[~]
└─$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.16.236.2             ether   00:50:56:e7:8a:c3   C                     eth0
```

- Metasploitable routing table and ARP cache:



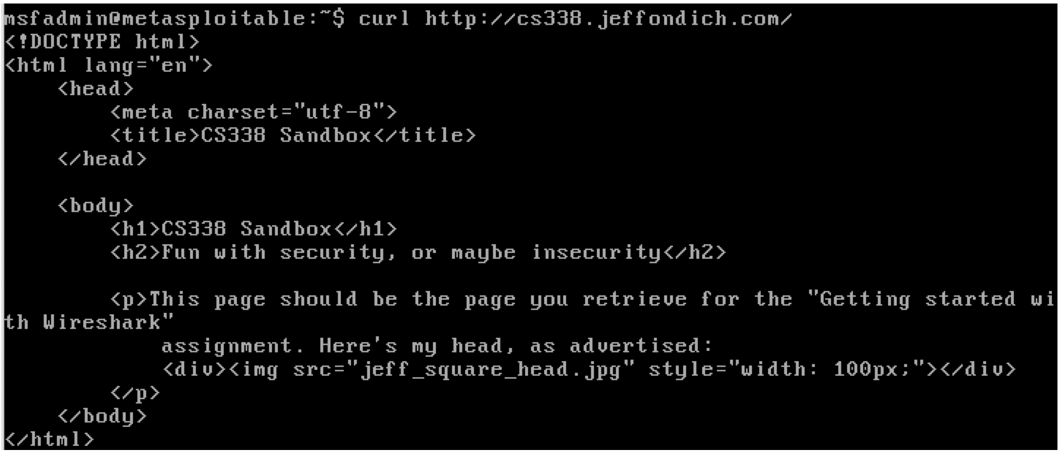
```
msfadmin@metasploitable:~$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
172.16.236.0    *               255.255.255.0   U         0 0          0 eth0
default         172.16.236.2    0.0.0.0         UG        0 0          0 eth0
msfadmin@metasploitable:~$ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
172.16.236.0    0.0.0.0         255.255.255.0   U         0 0          0 eth0
0.0.0.0         172.16.236.2    0.0.0.0         UG        0 0          0 eth0
msfadmin@metasploitable:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
172.16.236.2             ether   00:50:56:E7:8A:C3   C                     eth0
msfadmin@metasploitable:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.16.236.2             ether   00:50:56:E7:8A:C3   C                     eth0
```

- MAC address to which Metasploitable should send the TCP SYN packet: 00:50:56:E7:8A:C3
    - If the user of Metasploitable wants to get the CS338 sandbox page via the command `curl http://cs338.jeffondich.com/`. then Metasploitable should send the TCP SYN packet to the MAC address 00:50:56:E7:8A:C3. The IP address of http://cs338.jeffondich.com/ is 137.22.198.40 (non-authoritative answer 45.79.89.123). From the routing table, we see that the  IP address on the local network corresponding to the "first hop" for packets sent is 172.16.236.2. This is a gateway to 0.0.0.0, which is a gateway to 172.16.236.0. From the ARP cache, we see that the MAC address of 172.16.236.2 is 00:50:56:E7:8A:C3.

*Jeff's steps:*

- IP address of http://cs338.jeffondich.com/ → 137.22.198.40
    - non-authoritative answer → 45.79.89.123

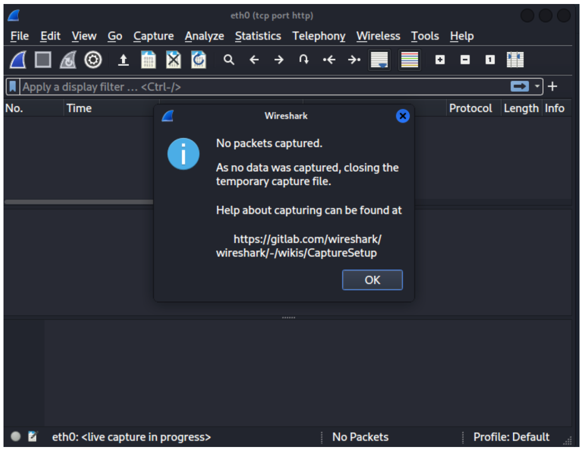
    - check routing table to determine which IP address on local network should be "first hop" for packets sent to 137.22.198.40 → 172.16.236.2
    - check ARP cache to determine MAC address of 172.16.236.2 → 00:50:56:E7:8A:C3
- capture packets for "tcp port http", execute "curl [http://cs338.jeffondich.com/](http://cs338.jeffondich.com/)" on Metasploitable →
    - yes, there is HTTP response on Metasploitable
    - no, there are no captured packets on Kali



```
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_head.jpg" style="width: 100px;"></div>
        </p>
    </body>
</html>
```



No packets captured.

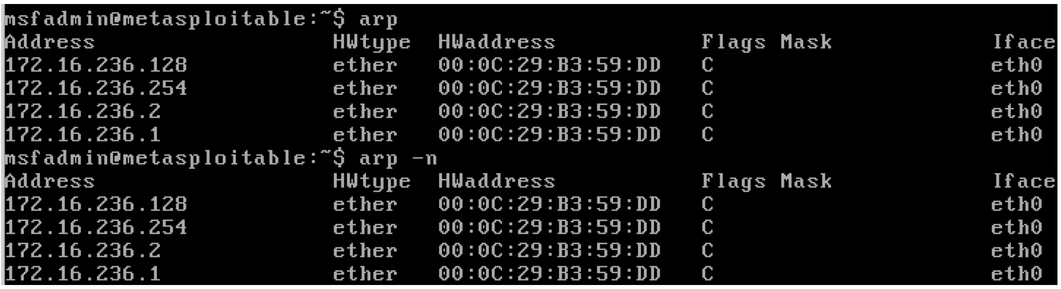
As no data was captured, closing the temporary capture file.

Help about capturing can be found at

https://gitlab.com/wireshark/wireshark/-/wikis/CaptureSetup

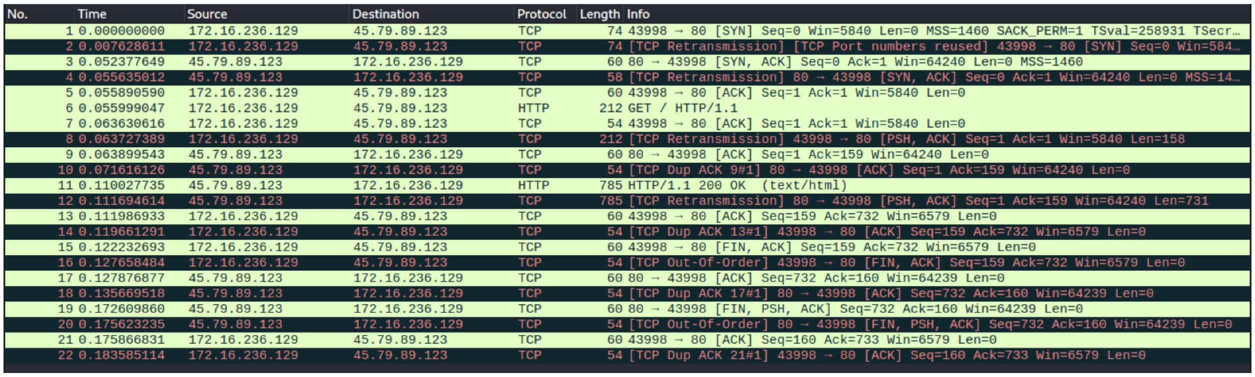
eth0: <live capture in progress> | No Packets | Profile: Default after Ettercap ARP poisoning:

- Metasploitable ARP cache:
    - previously contained only 172.16.236.2 (with MAC address 00:50:56:E7:8A:C3)
    - now contains 172.16.236.1, 172.16.236.2, 172.16.236.128, 172.16.236.254 (all with MAC address 00:0C:29:B3:59:DD)



```
msfadmin@metasploitable:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
172.16.236.128           ether   00:0C:29:B3:59:DD   C                     eth0
172.16.236.254           ether   00:0C:29:B3:59:DD   C                     eth0
172.16.236.2             ether   00:0C:29:B3:59:DD   C                     eth0
172.16.236.1             ether   00:0C:29:B3:59:DD   C                     eth0
msfadmin@metasploitable:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.16.236.128           ether   00:0C:29:B3:59:DD   C                     eth0
172.16.236.254           ether   00:0C:29:B3:59:DD   C                     eth0
172.16.236.2             ether   00:0C:29:B3:59:DD   C                     eth0
172.16.236.1             ether   00:0C:29:B3:59:DD   C                     eth0
```

- MAC address to which Metasploitable should send the TCP SYN packet: 00:0C:29:B3:59:DD
    - We still want packets sent to 172.16.236.2, and 00:0C:29:B3:59:DD is now the MAC address that corresponds to that IP address.
- capture packets for “tcp port http”, execute “curl http://cs338.jeffondich.com/” on Metasploitable →
    - yes, there is HTTP response on Metasploitable
    - yes, there are captured packets on Kali
        - “non-authoritative answer” IP address (45.79.89.123) is used
        - messages include:
            - TCP handshake
            - TCP “Retransmission”
            - TCP “Dup ACK”
            - TCP “Out-Of-Order”



| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 1 | 0.000000000 | 172.16.236.129 | 45.79.89.123 | TCP | 74 | 43998 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=258931 TSecr… |
| 2 | 0.007628611 | 172.16.236.129 | 45.79.89.123 | TCP | 74 | [TCP Retransmission] [TCP Port numbers reused] 43998 → 80 [SYN] Seq=0 Win=584… |
| 3 | 0.052377649 | 45.79.89.123 | 172.16.236.129 | TCP | 60 | 80 → 43998 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 |
| 4 | 0.055635012 | 45.79.89.123 | 172.16.236.129 | TCP | 58 | [TCP Retransmission] 80 → 43998 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=14… |
| 5 | 0.055890590 | 172.16.236.129 | 45.79.89.123 | TCP | 60 | 43998 → 80 [ACK] Seq=1 Ack=1 Win=5840 Len=0 |
| 6 | 0.055999047 | 172.16.236.129 | 45.79.89.123 | HTTP | 212 | GET / HTTP/1.1 |
| 7 | 0.063630616 | 172.16.236.129 | 45.79.89.123 | TCP | 54 | 43998 → 80 [ACK] Seq=1 Ack=1 Win=5840 Len=0 |
| 8 | 0.063727389 | 172.16.236.129 | 45.79.89.123 | TCP | 212 | [TCP Retransmission] 43998 → 80 [PSH, ACK] Seq=1 Ack=1 Win=5840 Len=158 |
| 9 | 0.063899543 | 45.79.89.123 | 172.16.236.129 | TCP | 60 | 80 → 43998 [ACK] Seq=1 Ack=159 Win=64240 Len=0 |
| 10 | 0.071616126 | 45.79.89.123 | 172.16.236.129 | TCP | 54 | [TCP Dup ACK 9#1] 80 → 43998 [ACK] Seq=1 Ack=159 Win=64240 Len=0 |
| 11 | 0.110027735 | 45.79.89.123 | 172.16.236.129 | HTTP | 785 | HTTP/1.1 200 OK  (text/html) |
| 12 | 0.111694614 | 45.79.89.123 | 172.16.236.129 | TCP | 785 | [TCP Retransmission] 80 → 43998 [PSH, ACK] Seq=1 Ack=159 Win=64240 Len=731 |
| 13 | 0.111986933 | 172.16.236.129 | 45.79.89.123 | TCP | 60 | 43998 → 80 [ACK] Seq=159 Ack=732 Win=6579 Len=0 |
| 14 | 0.119661291 | 172.16.236.129 | 45.79.89.123 | TCP | 54 | [TCP Dup ACK 13#1] 43998 → 80 [ACK] Seq=159 Ack=732 Win=6579 Len=0 |
| 15 | 0.122232693 | 172.16.236.129 | 45.79.89.123 | TCP | 60 | 43998 → 80 [FIN, ACK] Seq=159 Ack=732 Win=6579 Len=0 |
| 16 | 0.127658484 | 172.16.236.129 | 45.79.89.123 | TCP | 54 | [TCP Out-Of-Order] 43998 → 80 [FIN, ACK] Seq=159 Ack=732 Win=6579 Len=0 |
| 17 | 0.127876877 | 45.79.89.123 | 172.16.236.129 | TCP | 60 | 80 → 43998 [ACK] Seq=732 Ack=160 Win=64239 Len=0 |
| 18 | 0.135669518 | 45.79.89.123 | 172.16.236.129 | TCP | 54 | [TCP Dup ACK 17#1] 80 → 43998 [ACK] Seq=732 Ack=160 Win=64239 Len=0 |
| 19 | 0.172609860 | 45.79.89.123 | 172.16.236.129 | TCP | 60 | 80 → 43998 [FIN, PSH, ACK] Seq=732 Ack=160 Win=64239 Len=0 |
| 20 | 0.175623235 | 45.79.89.123 | 172.16.236.129 | TCP | 54 | [TCP Out-Of-Order] 80 → 43998 [FIN, PSH, ACK] Seq=732 Ack=160 Win=64239 Len=0 |
| 21 | 0.175866831 | 172.16.236.129 | 45.79.89.123 | TCP | 60 | 43998 → 80 [ACK] Seq=160 Ack=733 Win=6579 Len=0 |
| 22 | 0.183585114 | 172.16.236.129 | 45.79.89.123 | TCP | 54 | [TCP Dup ACK 21#1] 43998 → 80 [ACK] Seq=160 Ack=733 Win=6579 Len=0 |

explain:

- The TCP Out-Of-Order message can indicate that there are multiple paths between source and destination. This would explain why the ARP cache grew. The packets traveled to more IP addresses after the poisoning. [https://osqa-ask.wireshark.org/questions/1698/tcp-out-of-order-what-does-it-means/#:~:text=It%20simply%20means%20that%20particular,a%20through%20a%20longer%20path]
- The TCP Retransmission message and TCP Dup ACK can indicate that packets have been lost. The TCP Retransmission message is sent when an entity does not receive an acknowledgment of a packet before some timer expires. The TCP Dup ACK message is sent when the same ACK number is used to acknowledge different packets because there has been a gap in the sequence numbers. (If the packet with sequence number 5 is lost, then acknowledgement number 5 is sent for packets with sequence numbers 6,7,8, and so on, until packet 5 is received.) [https://accedian.com/blog/network-packet-loss-retransmissions-and-duplicate-acknowledgements/, https://www.cspsprotocol.com/tcp-retransmission/]
- It makes sense that these messages appear together. If a packet was lost, then we would expect retransmission (since no acknowledgement of lost packet), duplicate acknowledgement (since sequence order now has gap), and out-of-order packets (since missing packet will be received later than intended and after others).
- It seems that the ARP poisoning forced the packets to travel through different locations on the local network (the IP addresses all began with 172.16.236….). This led to loss of packets, and the stream of communication struggled.
- The poisoning also made the local network communicate with the "non-authoritative answer" IP address. The non-authoritative server does not contain the original files of the domain. It utilizes a cache of requested DNS records from all the DNS lookups done previously. [https://support.cpanel.net/hc/en-us/articles/360056527174-Authoritative-VS-Non-Authoritative-DNS-Servers]

create spoofing detector:

- From basic research (previous links), it seems that these TCP messages are not something that should cause great concern. This means that false positives of poisoning could definitely be an issue if we simply alerted entities any time these messages appeared.
- If I were to create a spoofing detector, I would check the frequency of these messages and compare to a threshold (below which is not concerning or alarming).
- I would also try to track the number of locations on the local network that the packets travel through. This information should be available in the routing table or ARP cache. If packets are sent through many gateways, then that may be worrisome and offers more opportunities for packets to get lost.
- I would also check if the IP address of the domain is the authoritative or non-authoritative IP address. It seems that the authoritative server should be used because it is responsible for proper record maintenance and responses. The non-authoritative server seems less reliable, as it uses a cache.