Kevin Clelland and Eric Gassel

a) What is Kali's main interface's MAC address? (The main interface is probably called eth0, but check ifconfig to be sure.)

08:00:27:76:7c:95

b) What is Kali's main interface's IP address?

137.22.174.250

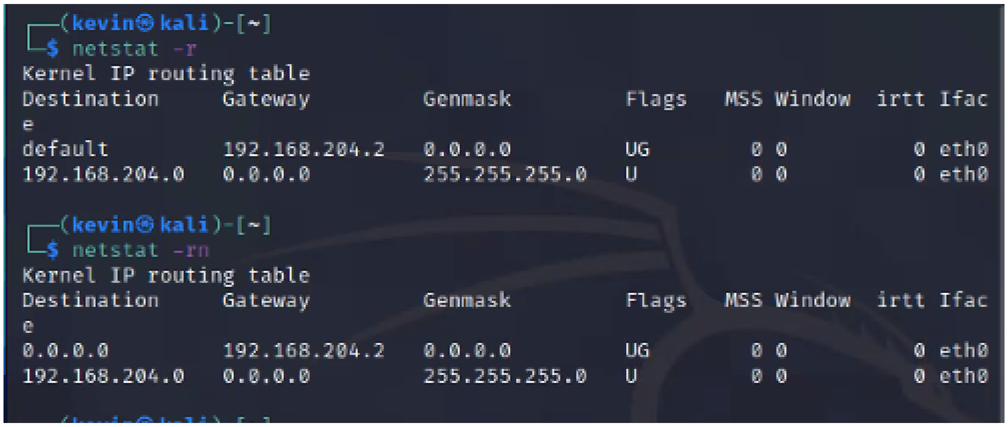
c) What is Metasploitable's main interface's MAC address?

08:00:27:d5:b7:15

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

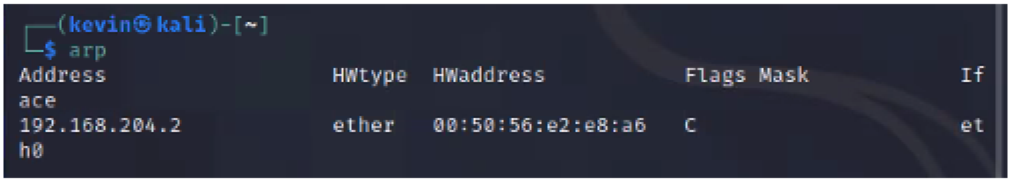
127.0.1.1

e) Show Kali's routing table. (Use "netstat -r" to see it with symbolic names, or "netstat -rn" to see it with numerical addresses.)



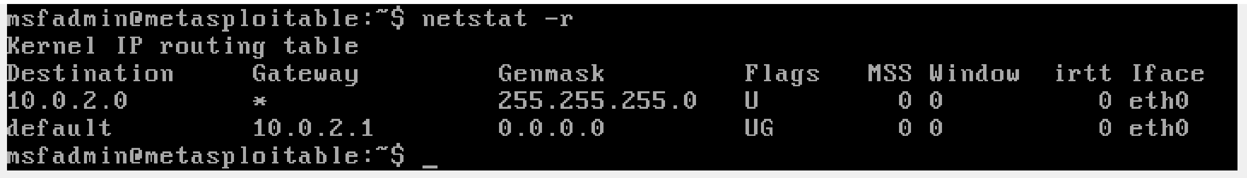
```
┌──(kevin㉿kali)-[~]
└─$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Ifac
e
default         192.168.204.2   0.0.0.0         UG        0 0          0 eth0
192.168.204.0   0.0.0.0         255.255.255.0   U         0 0          0 eth0

┌──(kevin㉿kali)-[~]
└─$ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Ifac
e
0.0.0.0         192.168.204.2   0.0.0.0         UG        0 0          0 eth0
192.168.204.0   0.0.0.0         255.255.255.0   U         0 0          0 eth0
```

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



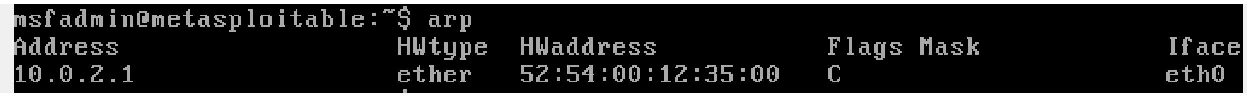
```
┌──(kevin㉿kali)-[~]
└─$ arp
Address                  HWtype  HWaddress           Flags Mask            If
ace
192.168.204.2            ether   00:50:56:e2:e8:a6   C                     et
h0
```

g) Show Metasploitable's routing table.



```
msfadmin@metasploitable:~$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.2.0        *               255.255.255.0   U         0 0          0 eth0
default         10.0.2.1        0.0.0.0         UG        0 0          0 eth0
msfadmin@metasploitable:~$ _
```

h) Show Metasploitable's ARP cache.



```
msfadmin@metasploitable:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.1                 ether   52:54:00:12:35:00   C                     eth0
```

i) Suppose the user of Metasploitable wants to get the CS231 sandbox page via the command "curl http://cs231.jeffondich.com/". To which MAC address should Metasploitable send the TCP SYN packet to get the whole HTTP query started? Explain why.

52:54:00:12:35:00. That's the MAC address associated with our IP address (before hacking occurs). In g, we see that our default IP address is 10.0.2.1, so this is the IP address of the server/browser we are working on. And in h, we see that 52:54:00:12:35:00 is the hardware address associated with that IP address.

j) Fire up Wireshark on Kali. Start capturing packets for "tcp port http". On Metasploitable, execute "curl http://cs231.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 get nothing in Wireshark, which is expected because we aren't actually eavesdropping on that connection yet. We do see an http response on metasploitable.

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:

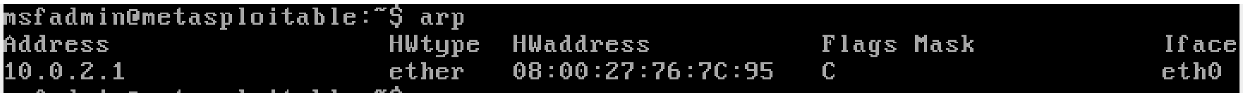
   i) Start sniffing (*not* bridged sniffing) on eth0
   ii) Scan for Hosts
   iii) View the Hosts list
   iv) Select your Metasploit VM from the Host List
   v) Add that host as Target 1
   vi) Start ARP Poisoning (including Sniff Remote Connections)
   vii) Do your stuff with wireshark and Metasploit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

This time we did capture packets on the kali machine (and we got an http response on metasploitable).

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

The HWaddress changed and is now directed to the kali machine's MAC address (which is good because ettercap is doing a man in the middle attack)



```
msfadmin@metasploitable:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.1                 ether   08:00:27:76:7C:95   C                     eth0
```

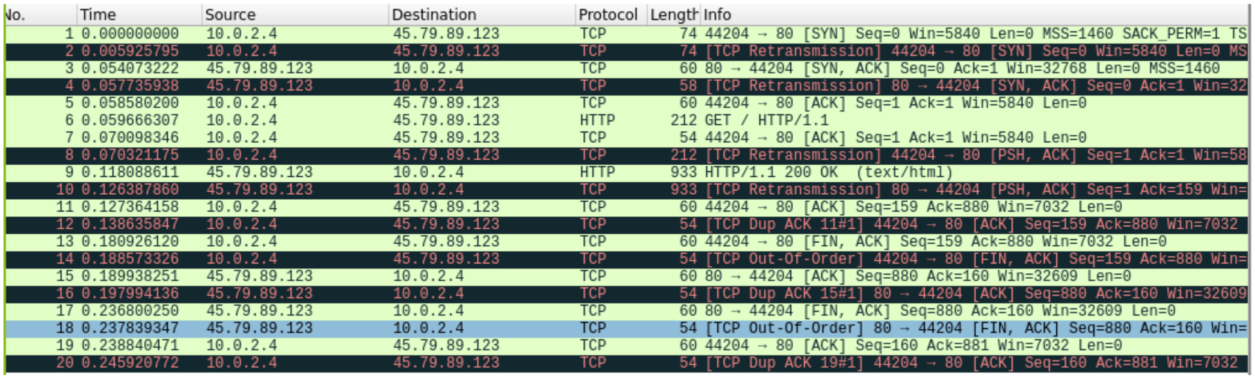
m) If you execute "curl http://cs231.jeffondich.com/" on Metasploitable now, to what MAC address will Metasploitable send the TCP SYN packet? Explain why.

It will send it to the kali machine because that is now operating as the man in the middle.

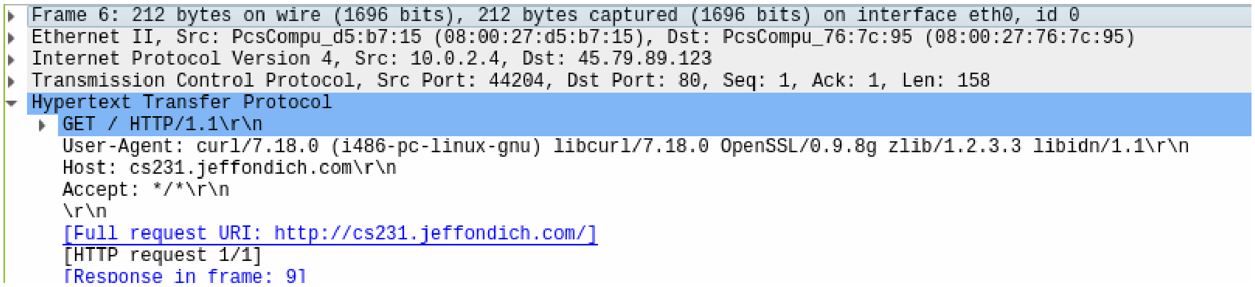
n) Start Wireshark capturing "tcp port http" again.

o) Execute "curl http://cs231.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 cs231.jeffondich.com?

Yes we saw an HTTP response in metasploitable. And we have captured packets in wireshark with the same information (and it's not encrypted because it's HTTP not HTTPS, so we can tell what messages went back and forth). For example, the second picture below shows a GET request sent to the server. We also know we're seeing messages from both the metasploitable machine and the server because we're seeing messages sent to and from both IP addresses in wireshark.



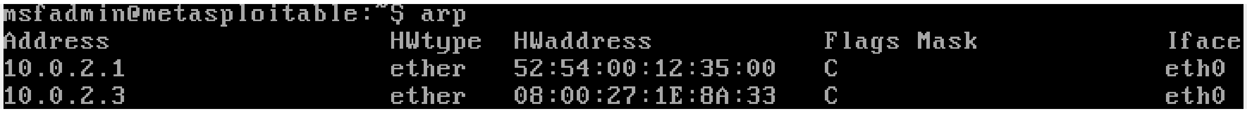
```
No.   Time          Source          Destination     Protocol  Length Info
    1 0.000000000   10.0.2.4        45.79.89.123    TCP          74 44204 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TS
    2 0.005925795   10.0.2.4        45.79.89.123    TCP          74 [TCP Retransmission] 44204 → 80 [SYN] Seq=0 Win=5840 Len=0 MS
    3 0.054073222   45.79.89.123    10.0.2.4        TCP          60 80 → 44204 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1460
    4 0.057735938   45.79.89.123    10.0.2.4        TCP          58 [TCP Retransmission] 80 → 44204 [SYN, ACK] Seq=0 Ack=1 Win=32
    5 0.058580200   10.0.2.4        45.79.89.123    TCP          60 44204 → 80 [ACK] Seq=1 Ack=1 Win=5840 Len=0
    6 0.059666307   10.0.2.4        45.79.89.123    HTTP        212 GET / HTTP/1.1
    7 0.070098346   10.0.2.4        45.79.89.123    TCP          54 44204 → 80 [ACK] Seq=1 Ack=1 Win=5840 Len=0
    8 0.070321175   10.0.2.4        45.79.89.123    TCP         212 [TCP Retransmission] 44204 → 80 [PSH, ACK] Seq=1 Ack=1 Win=58
    9 0.118088611   45.79.89.123    10.0.2.4        HTTP        933 HTTP/1.1 200 OK  (text/html)
   10 0.126387860   45.79.89.123    10.0.2.4        TCP         933 [TCP Retransmission] 80 → 44204 [PSH, ACK] Seq=1 Ack=159 Win=
   11 0.127364158   10.0.2.4        45.79.89.123    TCP          60 44204 → 80 [ACK] Seq=159 Ack=880 Win=7032 Len=0
   12 0.138635847   10.0.2.4        45.79.89.123    TCP          54 [TCP Dup ACK 11#1] 44204 → 80 [ACK] Seq=159 Ack=880 Win=7032
   13 0.180926120   10.0.2.4        45.79.89.123    TCP          60 44204 → 80 [FIN, ACK] Seq=159 Ack=880 Win=7032 Len=0
   14 0.188573326   10.0.2.4        45.79.89.123    TCP          54 [TCP Out-Of-Order] 44204 → 80 [FIN, ACK] Seq=159 Ack=880 Win=
   15 0.189938251   45.79.89.123    10.0.2.4        TCP          60 80 → 44204 [ACK] Seq=880 Ack=160 Win=32609 Len=0
   16 0.197994136   45.79.89.123    10.0.2.4        TCP          54 [TCP Dup ACK 15#1] 80 → 44204 [ACK] Seq=880 Ack=160 Win=32609
   17 0.236800250   45.79.89.123    10.0.2.4        TCP          60 80 → 44204 [FIN, ACK] Seq=880 Ack=160 Win=32609 Len=0
   18 0.237839347   45.79.89.123    10.0.2.4        TCP          54 [TCP Out-Of-Order] 80 → 44204 [FIN, ACK] Seq=880 Ack=160 Win=
   19 0.238840471   10.0.2.4        45.79.89.123    TCP          60 44204 → 80 [ACK] Seq=160 Ack=881 Win=7032 Len=0
   20 0.245920772   10.0.2.4        45.79.89.123    TCP          54 [TCP Dup ACK 19#1] 44204 → 80 [ACK] Seq=160 Ack=881 Win=7032
```



```
▸ Frame 6: 212 bytes on wire (1696 bits), 212 bytes captured (1696 bits) on interface eth0, id 0
▸ Ethernet II, Src: PcsCompu_d5:b7:15 (08:00:27:d5:b7:15), Dst: PcsCompu_76:7c:95 (08:00:27:76:7c:95)
▸ Internet Protocol Version 4, Src: 10.0.2.4, Dst: 45.79.89.123
▸ Transmission Control Protocol, Src Port: 44204, Dst Port: 80, Seq: 1, Ack: 1, Len: 158
▾ Hypertext Transfer Protocol
  ▸ GET / HTTP/1.1\r\n
    User-Agent: curl/7.18.0 (i486-pc-linux-gnu) libcurl/7.18.0 OpenSSL/0.9.8g zlib/1.2.3.3 libidn/1.1\r\n
    Host: cs231.jeffondich.com\r\n
    Accept: */*\r\n
    \r\n
    [Full request URI: http://cs231.jeffondich.com/]
    [HTTP request 1/1]
    [Response in frame: 9]
```

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/Ettercap got into that host’s ARP cache and changed the target MAC address that Metasploitable thinks corresponds to its own internet address. So instead of sending the packets/requests to it’s (virtual) network adapter, it’s sending them over to the kali machine’s (virtual) adapter, which is then passing them on and returning the input (acting as a man in the middle). So the metasploitable machine doesn’t know about the spoofing, but the Kali machine can listen to it’s own tcp port and sniff all the packets being sent to and from the metasploitable machine and the server. Essentially, Kali is inserting itself as the ‘router’.



```
msfadmin@metasploitable:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.1                 ether   52:54:00:12:35:00   C                     eth0
10.0.2.3                 ether   08:00:27:1E:8A:33   C                     eth0
```

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.)

The ARP spoofing detector would need to confirm that the MAC address (HWaddress) is the correct one, and that it does in fact point toward the desired server. It would also need to check that the MAC address matches the desired IP address. To implement this, you could maybe store this address in the machine somewhere else, so it could be checked against before making a query/request. You could also ask the server to prove it’s identity using some kind of public key or certificate, although that might require some new functionality on the adapter’s part, and could be difficult to expect of all your computer’s hardware.