John Woods

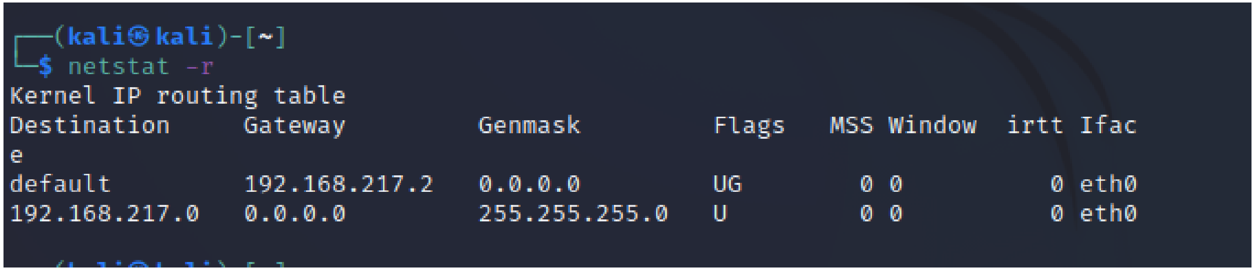
a. Kali MAC: 00:0c:29:1c:6e:1e

b. Kali ip: 192.168.217.128

c. Metasploitable MAC: 00:0c:29:1a:c3:4f

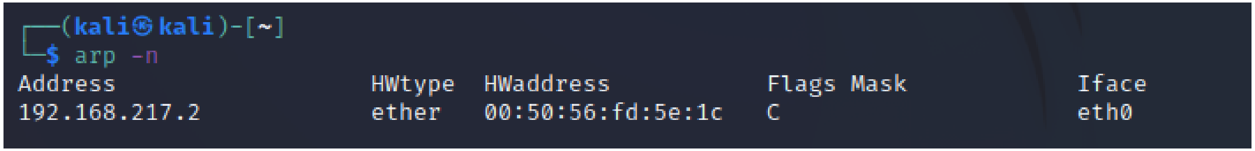
d. Metasploitable IP: 192.168.217.130

e.



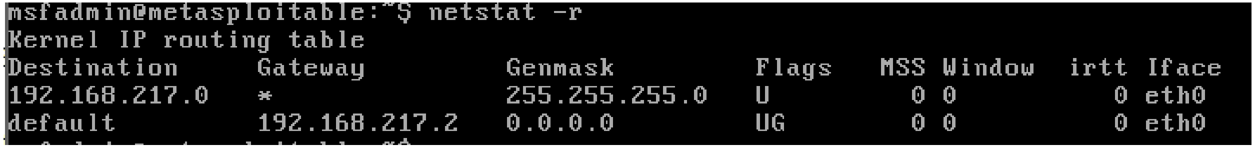
```
┌──(kali㉿kali)-[~]
└─$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Ifac
e
default         192.168.217.2   0.0.0.0         UG        0 0          0 eth0
192.168.217.0   0.0.0.0         255.255.255.0   U         0 0          0 eth0
```

f.



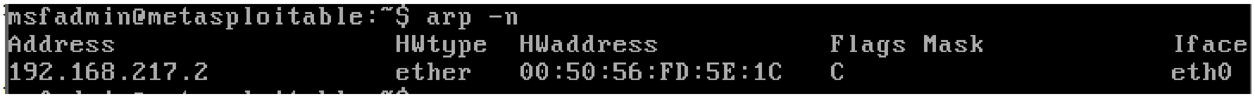
```
┌──(kali㉿kali)-[~]
└─$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.217.2            ether   00:50:56:fd:5e:1c   C                     eth0
```

g.



```
msfadmin@metasploitable:~$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.217.0   *               255.255.255.0   U         0 0          0 eth0
default         192.168.217.2   0.0.0.0         UG        0 0          0 eth0
```

h.



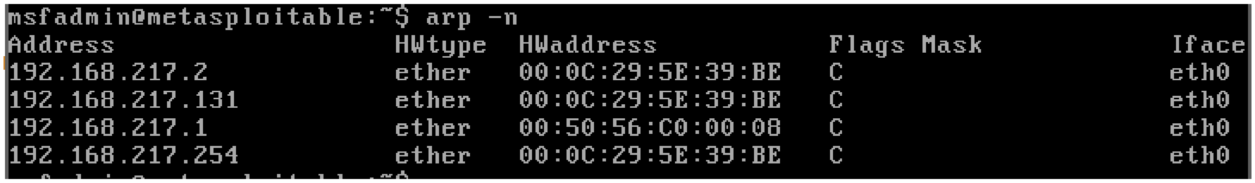
```
msfadmin@metasploitable:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.217.2            ether   00:50:56:FD:5E:1C   C                     eth0
```

i. The user should choose the MAC address of the result from the routing table, i.e. the result in the arp cache. So in this case, it should be 00:50:56:FD:5E:1C. This is because although we want to connect to jeffondich.com, we need to first send the packet to the first computer in the line so it can continue downwards towards its destination.

j. I do see an HTTP response in Metasploitable, it seems to just be the actual raw text of the http. I also see captured packets in wireshark, documenting the TCP handshake as well as the HTTP call and response.

k. Check!

l.



```
msfadmin@metasploitable:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.217.2            ether   00:0C:29:5E:39:BE   C                     eth0
192.168.217.131          ether   00:0C:29:5E:39:BE   C                     eth0
192.168.217.1            ether   00:50:56:C0:00:08   C                     eth0
192.168.217.254          ether   00:0C:29:5E:39:BE   C                     eth0
```

This arp cache is much larger than before, and includes some addresses we haven't seen before. This is likely due to my arp poisoning attack.

m. I imagine that metasploitable will send the TCP SYN request to kali rather than straight to the correct location, as kali has managed to ARP poison it's way into being seen as the correct address (notice how 192.168.217.131 shares the same MAC address as the IP above it!).

n. Check!

o. I do see the HTTP response in metasploitable still, and I also see captured packets in wireshark. This time however, I can also tell what messages went between Metasploitable and jeff’s website!

p. Essentially, Kali seems to be claiming the MAC address of the destination computer as its own, constantly bombarding metasploitable with messages claiming that it’s got both it’s own MAC address as well as the address of the outbound machine. As such, when Metasploitable wraps its message in the ethernet frame with the mac address in question, it actually ends up being received by both the destination machine as well as kali. This process of listening for arp requests from the target machine and then responding is the core of arp poisoning.

q. If I wanted to design an arp spoofing detector, I’d probably have it look for cases where we have an IP address that’s using two MAC addresses, as we see highlighted in yellow within wireshark. While this would likely detect problems, we might also get false positives if IP addresses shift since they’re not permanent, and our old ARP table might be populated with outdated data, giving a false positive. Similarly, if we design a detector designed to see if a mac address is held by two different IP’s, we might tun into the same issue since IP addresses are impermanent.