McKenna Wirth and Antonia Ritter
CS338 Spring 2022
PERSON-IN-THE-MIDDLE VIA ARP SPOOFING

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

   00:0c:29:0a:17:b8

B. *What is Kali's main interface's IP address?*

   192.168.249.128

C. *What is Metasploitable's main interface's MAC address?*

   00:0c:29:b9:8a:17

D. *What is Metasploitable's main interface's IP address?*

   192.168.249.128

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

Kernel IP routing table

| Destination | Gateway | Genmask | Flags | MSS | Window | irtt | Iface |
|---|---|---|---|---|---|---|---|
| default | 192.168.249.2 | 0.0.0.0 | UG | 0 | 0 | 0 | eth0 |
| 192.168.249.0 | 0.0.0.0 | 255.255.255.0 | U | 0 | 0 | 0 | eth0 |

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

| Address | HWtype | HWaddress | Flags | Mask | Iface |
|---|---|---|---|---|---|
| 192.168.249.2 | ether | 00:50:56:fa:52:54 | C | | eth0 |
| 192.168.249.129 | ether | 00:0c:29:b9:da:17 | C | | eth0 |

*G. Show Metasploitable's routing table.*

| Destination | Gateway | Genmask | Flags | MSS | Window | irtt | Iface |
|---|---|---|---|---|---|---|---|
| 192.168.249.0 | 0.0.0.0 | 255.255.255.0 | U | 0 | 0 | 0 | eth0 |
| 0.0.0.0 | 192.168.249.2 | 0.0.0.0 | UG | 0 | 0 | 0 | eth0 |

*H. Show Metasploitable's ARP cache.*

| Address | HWtype | HWaddress | Flags | Mask | Iface |
|---|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| 192.168.249.2 | ether | 00:50:56:FA:52:54 | C | eth0 |
| 192.168.249.128 | ether | 00:0C:29:0A:17:B8 | C | eth0 |

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.*

It needs to first send the packet to the gateway, which has IP address 192.168.249.2, corresponding to MAC address 00:50:56:FA:52:54, as seen in the ARP cache.

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 saw an HTTP response on Metasploitable - the html for this page. In Wireshark we have 11 captured packets, consisting of TCP handshake and HTTP request and information. TCP acknowledgements are sent and the connection is closed.

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:*

   a. *Start sniffing (not bridged sniffing) on eth0*
   b. *Scan for Hosts*
   c. *View the Hosts list*
   d. *Select your Metasploit VM from the Host List*
   e. *Add that host as Target 1*
   f. *Start ARP Poisoning (including Sniff Remote Connections)*
   g. *Do your stuff with wireshark and Metasploit*
   h. *Stop ARP Poisoning*
   i. *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?*

| Address | HWtype | HWaddress | Flags | Mask | Iface |
|---|---|---|---|---|---|
| 192.168.249.2 | ether | 00:0C:29:0A:17:B8 | C | | eth0 |

| | | | | |
|---|---|---|---|---|
| 192.168.249.128 | ether | 00:0C:29:0A:17:B8 | C | eth0 |
| 192.168.249.1 | ether | 00:0C:29:0A:17:B8 | C | eth0 |
| 192.168.249.254 | ether | 00:0C:29:0A:17:B8 | C | eth0 |

All of the MAc addresses are now the same - the address for Kali, where Ettercap is running. This means that all traffic is going through Kali first.

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 it to the Kali MAC address, since that is what is listed in the ARP cache as corresponding to the gateway.

N. *Start Wireshark capturing "tcp port http" again.*

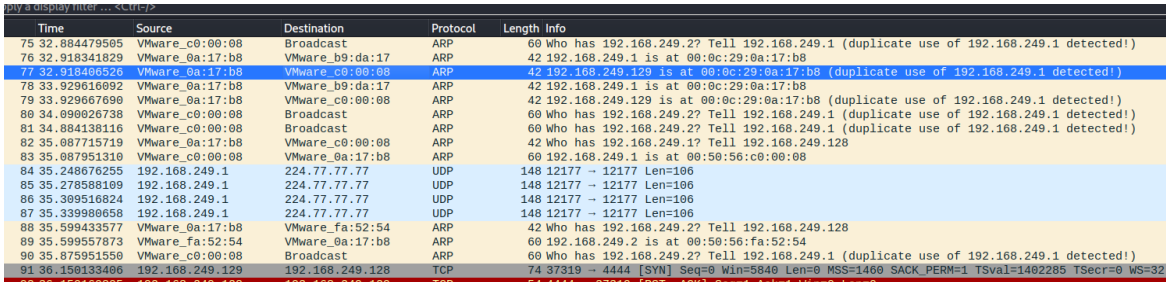
O. *Execute "curl http://cs338.jeffondich.com/" on Metasploitable. On Kali, stop capturing. Do you see an HTTP response on Metasploitable? Do you see captured packets in Wireshark? Can you tell from Kali what messages went back and forth between Metasploitable and cs338.jeffondich.com?*

We did see an HTTP response on Metasploitable. On Wireshark, there are twice as many as in part J. Every packet we saw earlier is duplicated. The first packet in each pair is sent from Metasploitable to Kali's MAC address. The second is sent from Kali's MAC address Jeff's. From Kali we can see all the traffic, including the content of cs338.jeffondich.com and Mal passing along all the packets.

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 changed all the MAC addresses in Metasploitable's ARP cache to Kali's MAC address, so from Metasploitable's perspective, when it wanted to send something to a particular IP address, the cache said to send it to Kali.

The mechanism for poisoning the ARP cache is by spamming the network with ARP packets that associate all IP addresses with the desired spoofed MAC address. (https://charlesreid1.com/wiki/MITM/Wired/ARP_Poisoning_with_Ettercap) We checked this by starting Wireshark (see below), then initiating the poisoning. We saw several ARP packets sent from Kali, updating MAC addresses for various IPs to Kali's MAC address.



| | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 75 | 32.884479505 | VMware_c0:00:08 | Broadcast | ARP | 60 | Who has 192.168.249.2? Tell 192.168.249.1 (duplicate use of 192.168.249.1 detected!) |
| 76 | 32.918341829 | VMware_0a:17:b8 | VMware_b9:da:17 | ARP | 42 | 192.168.249.1 is at 00:0c:29:0a:17:b8 |
| 77 | 32.918486526 | VMware_0a:17:b8 | VMware_c0:00:08 | ARP | 42 | 192.168.249.129 is at 00:0c:29:0a:17:b8 (duplicate use of 192.168.249.1 detected!) |
| 78 | 33.929616092 | VMware_0a:17:b8 | VMware_b9:da:17 | ARP | 42 | 192.168.249.1 is at 00:0c:29:0a:17:b8 |
| 79 | 33.929667690 | VMware_0a:17:b8 | VMware_c0:00:08 | ARP | 42 | 192.168.249.129 is at 00:0c:29:0a:17:b8 (duplicate use of 192.168.249.1 detected!) |
| 80 | 34.090026738 | VMware_c0:00:08 | Broadcast | ARP | 60 | Who has 192.168.249.2? Tell 192.168.249.1 (duplicate use of 192.168.249.1 detected!) |
| 81 | 34.884138116 | VMware_c0:00:08 | Broadcast | ARP | 60 | Who has 192.168.249.2? Tell 192.168.249.1 (duplicate use of 192.168.249.1 detected!) |
| 82 | 35.087715719 | VMware_0a:17:b8 | VMware_c0:00:08 | ARP | 42 | Who has 192.168.249.1? Tell 192.168.249.128 |
| 83 | 35.087951310 | VMware_c0:00:08 | VMware_0a:17:b8 | ARP | 60 | 192.168.249.1 is at 00:50:56:c0:00:08 |
| 84 | 35.248676255 | 192.168.249.1 | 224.77.77.77 | UDP | 148 | 12177 → 12177 Len=106 |
| 85 | 35.278588109 | 192.168.249.1 | 224.77.77.77 | UDP | 148 | 12177 → 12177 Len=106 |
| 86 | 35.309516824 | 192.168.249.1 | 224.77.77.77 | UDP | 148 | 12177 → 12177 Len=106 |
| 87 | 35.339980658 | 192.168.249.1 | 224.77.77.77 | UDP | 148 | 12177 → 12177 Len=106 |
| 88 | 35.599433577 | VMware_0a:17:b8 | VMware_fa:52:54 | ARP | 42 | Who has 192.168.249.2? Tell 192.168.249.128 |
| 89 | 35.599557873 | VMware_fa:52:54 | VMware_0a:17:b8 | ARP | 60 | 192.168.249.2 is at 00:50:56:fa:52:54 |
| 90 | 35.875951550 | VMware_c0:00:08 | Broadcast | ARP | 60 | Who has 192.168.249.2? Tell 192.168.249.1 (duplicate use of 192.168.249.1 detected!) |
| 91 | 36.150133406 | 192.168.249.129 | 192.168.249.128 | TCP | 74 | 37319 → 4444 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=1402285 TSecr=0 WS=32 |

ARP activity in Wireshark after initiating an ARP poisoning attack with Ettercap.

*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.)*

First, if there are multiple IPs in the ARP cache, check whether any of them have the same MAC address. This might generate false positives if this occurs benignly, for instance if the same computer has multiple IP addresses.

Another way to detect ARP spoofing is to watch for suspicious ARP activity; as described above, Kali sent many ARP packets very quickly, declaring multiple IPs all had the same MAC address. However, the challenge is determining how much ARP activity counts as suspicious. Set the threshold too low and you'll get false positives.