ARP Spoofing

- CEG 6430/4430 Cyber Network Security
- Junjie Zhang
- junjie.zhang@wright.edu
- Wright State University

ARP

The Address Resolution Protocol (ARP) is a communication protocol used for discovering the link layer address, such as a MAC address, associated with a given internet layer address, typically an IPv4 address.

Display ARP Cache

You can use `arp -n` to display the ARP cache on a Linux system.

```
[jzhang@DESKTOP-DSVPHPI ARPSpoofing]$arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.31.96.1              ether   00:15:5d:97:a7:d6   C                     eth0
```

Clean Up ARP Cache

You occasionally want to "force" the system to generate an ARP request (i.e., to capture ARP packets), you will most likely neeed to clean up the cache. Otherwise, the system will use information in the ARP cache if available, thereby generating no ARP requests.

Clean Up ARP Cache

- Option 1: to use `arp -d`

```
[jzhang@DESKTOP-DSVPHPI ARPSpoofing]$arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.31.96.1              ether   00:15:5d:97:a7:d6   C                     eth0
[jzhang@DESKTOP-DSVPHPI ARPSpoofing]$sudo arp -d 172.31.96.1
[jzhang@DESKTOP-DSVPHPI ARPSpoofing]$arp -n
```

Unfortunately, for `arp -d` you have to specify the IP of the entry you want to delete. Alternatively, you can also use

Clean Up ARP Cache

- Option 2: to use `ip -s -s neigh flush all`

```
[jzhang@DESKTOP-DSVPHPI ARPSpoofing]$arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.31.96.1              ether   00:15:5d:97:a7:d6   C                     eth0
[jzhang@DESKTOP-DSVPHPI ARPSpoofing]$sudo ip -s -s neigh flush all
172.31.96.1 dev eth0 lladdr 00:15:5d:97:a7:d6 used 87/87/48 probes 4 STALE

*** Round 1, deleting 1 entries ***
*** Flush is complete after 1 round ***
[jzhang@DESKTOP-DSVPHPI ARPSpoofing]$arp -n
```

Generate ARP Packets

Then you can ping an external IP or domain name, your system will generate ARP request to resolve the MAC of the gateway IP. You can caputre packets and meanwhile find the MAC address is resolved and cached.

Generate ARP Packets

```
[jzhang@DESKTOP-DSVPHPI system32]$ping www.cnn.com
PING cnn-tls.map.fastly.net (146.75.83.5) 56(84) bytes of data.
64 bytes from 146.75.83.5 (146.75.83.5): icmp_seq=1 ttl=54 time=26.5 ms
64 bytes from 146.75.83.5 (146.75.83.5): icmp_seq=2 ttl=54 time=32.3 ms
^C
--- cnn-tls.map.fastly.net ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1079ms
rtt min/avg/max/mdev = 26.493/29.415/32.337/2.922 ms
[jzhang@DESKTOP-DSVPHPI system32]$arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.31.96.1              ether   00:15:5d:97:a7:d6   C                     eth0
```

Reviewing ARP Protocol

Communication In A Network Segment

Host A sends a packet to Host B; A knows B's IP address

https://github.com/jzhang369/images/blob/main/switch.png

Communication In A Network Segment

- A uses its IP address, the netmask, and B's IP address to know that A and B are in the same network segment. Therefore, B is reachable via one-hop communication.
- Since A does not know B's MAC address, it will send a a packet to the switch:
  - [Dst_MAC:FF:FF:FF:FF:FF:FF|Src_MAC:AAAA|who has 192.168.1.6]

Communication In A Network Segment

- The switch receives this packet.
  - It knows the packet is coming from AAAA via Interface 1, adding an entry [1|AAAA] to the table.
  - The dest MAC address is FF...FF, a broadcast address. So it floods the packet to all occupied interfaces except for Interface 1.

Communication In A Network Segment

- Host C receives this packet but drops it after parsing.
- Host D receives this packet but drops it after parsing.
- Host B receives the packet, parses it, and finds that the destination IP address matches with its own IP address. It will send a packet to the switch:
  - [Dst_MAC:AAAA|Src_MAC:BBBB|I have 192.168.1.6]

Communication In A Network Segment

- The switch receives this packet
  - It knows the packet is coming from BBBB via Interface 2, adding an entry [2|BBBB] to the table.
  - The dest MAC address is AAAA, which is associated with Interface 1. So it forwards the packet to Interface 1.

Communication In A Network Segment

- Host A receives this packet
  - It knows 192.168.1.6 has MAC of BBBB and keeps this mapping in its cache.
  - It creates an IP packet
    - [Dst_MAC:BBBB|Src_MAC:AAAA|Dst_IP:192.168.1.6|Src_IP: 192.168.1.5|Data]
  - It sends this packet to the switch.

Communication In A Network Segment

- The switch receives this packet
  - The dest MAC address is BBBB, which is associated with Interface 2. So it forwards the packet to Interface 2.
- Host B recevies this IP packet.

An Attack Scenario

Host A sends a packet to Host B; A knows B's IP address

https://github.com/jzhang369/images/blob/main/switch.png

Communication In A Network Segment

- A uses its IP address, the netmask, and B's IP address to know that A and B are in the same network segment. Therefore, B is reachable via one-hop communication.
- Since A does not know B's MAC address, it will send a a packet to the switch:
  - [Dst_MAC:FF:FF:FF:FF:FF:FF|Src_MAC:AAAA|who has 192.168.1.6]

Communication In A Network Segment

- The switch receives this packet.
  - It knows the packet is coming from AAAA via Interface 1, adding an entry [1|AAAA] to the table.
  - The dest MAC address is FF...FF, a broadcast address. So it floods the packet to all occupied interfaces except for Interface 1.

Communication In A Network Segment

- Host C receives this packet ~~but drops it after parsing~~. and responds with a packe to the switch:
  - P1: [Dst_MAC:AAAA|Src_MAC:CCCC|I have 192.168.1.6]
- Host D receives this packet but drops it after parsing.
- Host B receives the packet, parses it, and finds that the destination IP address matches with its own IP address. It will send a packet to the switch:
  - P2: [Dst_MAC:AAAA|Src_MAC:BBBB|I have 192.168.1.6]

Communication In A Network Segment

- The switch receives P1 and P2
  - It knows P1 is coming from CCCC via Interface 2, adding an entry [3|CCCC] to the table.
  - It forwards P1 to Interface 1.
  - It knows P2 is coming from BBBB via Interface 2, adding an entry [2|BBBB] to the table.
  - It forwards P2 to Interface 1.

Communication In A Network Segment

- Host A receives P1
  - It knows 192.168.1.6 has MAC of CCCC and keeps this mapping in its cache.
  - It creates an IP packet, namely P3
    - [Dst_MAC:CCCC|Src_MAC:AAAA|Dst_IP:192.168.1.6|Src_IP: 192.168.1.5|Data]
  - It sends this packet to the switch.
- Host A receives P2
  - The mapping <192.168.1.6, CCCC> has already been in the cache so P2 is dicarded

A Few Questions

- Who will receive P3 now?
- Can an attacker impersonate the IP of the gateway? Say (192.168.1.1)
- What should the attacker do with the intercepted packet?
- Which "direction" of session can you intercept?

A Few Questions

- Who will receive P3 now?
  - Host C.
  - This is the ARP Spoofing attack.

A Few Questions

- Can an attacker impersonate the IP of the gateway?
  - Yes and it is the common target.
  - If successful, the attacker will have the potential to intercept (1/2 of) the communication between the victim and the gateway.

A Few Questions

- What should the attacker do with the intercepted packet?
  - keep and then drop: interrupt the communication after sniffing the first packet, which is usually useless.
  - keep and forward: keep the communication going by fowarding this packet to its actual destination MAC.
- You can use `bettercap` to accomplish this.

```
$bettercap --sniffer --spoof ARP --proxy-http --proxy-https --custom-parser password
```

A Few Question

- Which "direction" of session can you intercept?
  - Assume the attacker, Host C, has successfully attack Host A and proxy the traffic from Host A to Host B.
    - Can attacker see packets from Host A to Host B?
      - Yes and this is the attack about.
    - Can attacker see packets from Host B to Host A?
      - No if it does not attack Host B. Host B will find Host A's MAC address independently.

Lab

Let's analyze a pcap trace using Wireshark.

- The original source of the trace is https://cs155.stanford.edu/hw_and_proj/proj3/sample.pcap.gz (description)
- A copy can also be found here.

Lab

Question 0: What is (likely) the subnet?

Question 1: Where was the trace collected? On an host or via the switch?

Question 2: Is there any ARP spoofing attack?

Question 3: Is there any successful ARP spoofing attack?

Question 4: If there is any successful ARP spoofing attack, can the attacker sniff all packets of the communication between the victim and its connected external host?

Lab

Question 0: What is (likely) the subnet?

- Use a filter `arp` in Wireshark, you will find a lot of ARP requests and responses, including spoofed ones.
  - You will find 128.3.23.1, which is likely to be the gateway's (router's) IP address. Its MAC: 80:0b:98:3b:b9:ec
  - The IPs concerned by these ARP queries and responses are 128.3.23.XXX. So the network segment is likely to be 128.3.23.0/24. (But possibly larger.)

Lab

Question 1: Where was the trace collected? On an host or via the switch?

- Use a filter `tcp` in Wireshark, you will find a number of IPs in this subnet are involved in established TCP connections.
  - 128.3.23.123, 128.3.23.227, 128.3.23.231, 128.3.23.245,
- Essentially the trace has the visiability of these IPs, so it should be collected from the mirorring port of the switch.

Lab

Question 2: Is there any ARP spoofing attack?

- Use a filter `arp` and you will find
  - 7c:d1:c3:94:9e:b8 claims it is the owner of multiple IPs.
  - 14:4f:8a:ed:c2:5e claims it is the owner of multiple IPs.
- A more effective way is to write a script for detection.

Lab

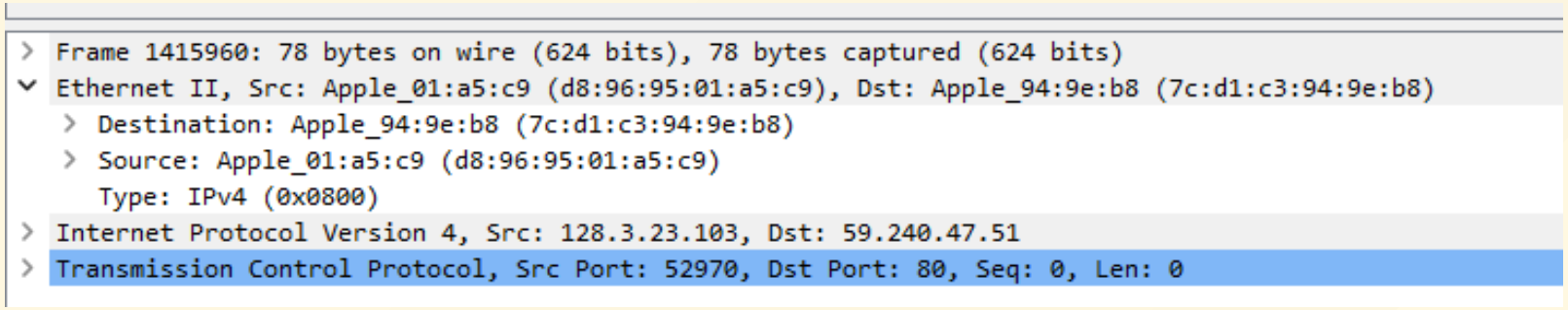
Question 3: Is there any successful ARP spoofing attack?

- Use this filter to detect connections that went to external networks but the destination MAC is 7c:d1:c3:94:9e:b8, which however should be the MAC of the gateway, 80:0b:98:3b:b9:ec.
  - tcp and eth.dst == 7c:d1:c3:94:9e:b8 and not(ip.dst == 128.3.23.0/24)
- You will find many TCP packets and 128.3.23.103 is an victim.

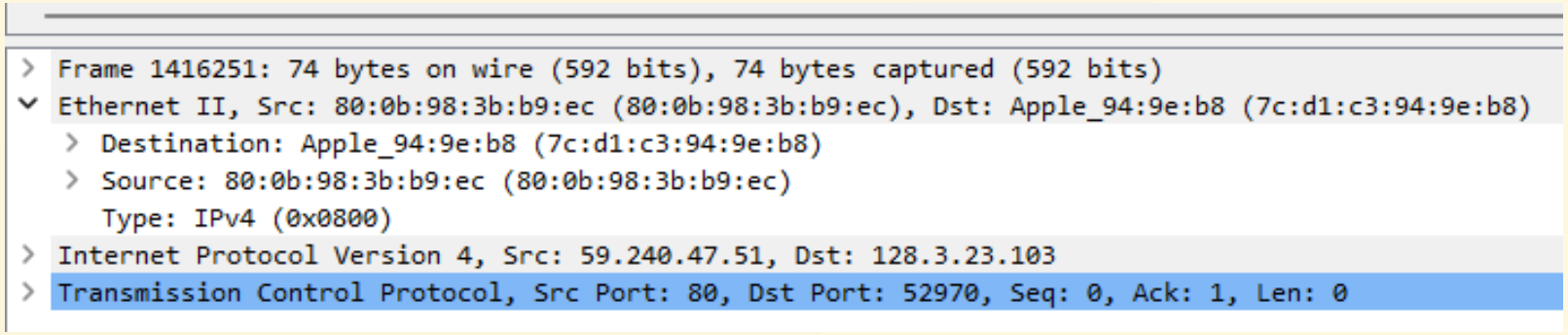
Lab

Question 4: If there is any successful ARP spoofing attack, can the attacker sniff all packets of the communication between the victim and its connected external host?

- Pick one of the packet (e.g., 1415960) identified for Question 3 and follow its TCP stream.
- Pkt 1415960: Src_MAC: d8:96:95:01:a5:c9, Dst_MAC:7c:d1:c3:94:9e:b8
  - 128.3.23.103 believes that 7c:d1:c3:94:9e:b8 is the gateway.



https://github.com/jzhang369/images/blob/main/arp1.png



https://github.com/jzhang369/images/blob/main/arp2.png

Lab

- Pkt 1415962: Src_MAC: 80:0b:98:3b:b9:ec, Dst_MAC:7c:d1:c3:94:9e:b8
  - 128.3.23.1 believes that 7c:d1:c3:94:9e:b8 is the MAC of 128.3.23.103.

So the answer is **yes**, where packets for both directions of this session are through the proxy of the attacker.

Essentially the attacker successfully attack both 128.3.23.103 and its gateway 128.3.23.1, as well as proxing their packets.