Zachary Kaplan
CS356
Ethernet & ARP Lab
4/13/17

```
Frame 48: 434 bytes on wire (3472 bits), 434 bytes captured (3472 bits)
on interface 0
Ethernet II, Src: HonHaiPr_59:00:19 (34:68:95:59:00:19), Dst:
Cisco_77:e4:00 (00:25:46:77:e4:00)
      Destination: Cisco_77:e4:00 (00:25:46:77:e4:00)
      Source: HonHaiPr_59:00:19 (34:68:95:59:00:19)
      Type: IPv4 (0x0800)
Data (420 bytes)

0000  45 00 01 a4 34 d0 40 00 40 06 b6 65 0a c8 cd d2
E...4.@.@..e....
0010  80 77 f5 0c cd 4a 00 50 0c f9 67 e6 b6 6b 3b 1b
.w...J.P..g..k;.
0020  80 18 00 e5 4f b5 00 00 01 01 08 0a 00 07 53 2f
....O.........S/
0030  43 4c ea ab 47 45 54 20 2f 77 69 72 65 73 68 61   CL..GET
/wiresha
0040  72 6b 2d 6c 61 62 73 2f 48 54 54 50 2d 65 74 68
rk-labs/HTTP-eth
0050  65 72 65 61                                  ....
```

1. The source Ethernet address of my device is 34:68:95:59:00:19.
2. The destination Ethernet address of this packet (the GET packet) is 00:25:46:77:e4:00. This address is not the ethernet address of the server that the requested page is on, but is instead the address of the gateway router my computer is connected to.
3. The 16-bit frame type identifier for this packet is 0x0800, which corresponds to IPv4.
4. The G appears at byte number 0x34 in the data section of the packet, or byte 52 in decimal. The data section immediately follows the ethernet frame, and so the offset of the G is exactly 52 bytes.

```
Frame 50: 1417 bytes on wire (11336 bits), 1417 bytes captured (11336
bits) on interface 0
Ethernet II, Src: Cisco_77:e4:00 (00:25:46:77:e4:00), Dst:
HonHaiPr_59:00:19 (34:68:95:59:00:19)
      Destination: HonHaiPr_59:00:19 (34:68:95:59:00:19)
      Source: Cisco_77:e4:00 (00:25:46:77:e4:00)
      Type: IPv4 (0x0800)
Data (1403 bytes)
```

```
0000  45 00 05 7b e1 57 40 00 34 06 12 07 80 77 f5 0c
E..{.W@.4....w..
0010  0a c8 cd d2 00 50 cd 4a b6 6b 3b 1b 0c f9 69 56
.....P.J.k;...iV
0020  80 10 00 eb 58 a5 00 00 01 01 08 0a 43 4c ea b9
....X.......CL..
0030  00 07 53 2f 48 54 54 50 2f 31 2e 31 20 32 30 30   ..S/HTTP/1.1
200
0040  20 4f 4b 0d 0a 44 61 74 65 3a 20 54 68 75 2c 20   OK..Date: Thu,
0050  31 33 20 41                                       ....
```

5. The source ethernet address is `00:25:46:77:e4:00`, this is again the address of the gateway router for my computer rather than the ethernet of the server the response is coming from.
6. The destination ethernet address of the packet (my computer's address) is `34:68:95:59:00:19`.
7. The Frame Type identifier is again `0x0800` in hex, and this represents `IPv4`.
8. The ASCII `O` appears at byte offset `0x41` in the data section of the packet. Therefore the O is exactly 65 bytes after the Ethernet frame.

```
Address       HWtype  HWaddress           Flags Mask            Iface
10.200.204.1  ether   00:25:46:77:e4:00   C                     wlan0
```

9. The address column indicates the cached IPv4 address , the HWtype column indicates that the cached hardware address is of type ethernet, the HWaddress column specifies the cached hardware address, the Flags column indicates the status of the cache entry (C indicates complete), and finally the Iface column shows what interface (ifconfig) the cache entry is relevant to.

```
      Frame 20: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on
interface 0
      Ethernet II, Src: HonHaiPr_59:00:19 (34:68:95:59:00:19), Dst: Broadcast
      (ff:ff:ff:ff:ff:ff)
            Destination: Broadcast (ff:ff:ff:ff:ff:ff)
            Source: HonHaiPr_59:00:19 (34:68:95:59:00:19)
            Type: ARP (0x0806)
      Address Resolution Protocol (request)
            Hardware type: Ethernet (1)
            Protocol type: IPv4 (0x0800)
            Hardware size: 6
            Protocol size: 4
            Opcode: request (1)
            Sender MAC address: HonHaiPr_59:00:19 (34:68:95:59:00:19)
```

```
        Sender IP address: 10.200.205.210
        Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
        Target IP address: 10.200.204.1
```

10. The destination address of the packet was broadcast or `ff:ff:ff:ff:ff:ff`, while the source address was my computer's ethernet address, `34:68:95:59:00:19`.
11. The 16-bit protocol identifier was `0x0806`, corresponding to `ARP`.
12.
    a. The ARP opcode field begins 7 bytes after the end of the ethernet frame.
    b. The ARP opcode for a request is 0x0001.
    c. Yes, the sender IP address is specified immediately after the sender MAC address.
    d. The final 4 bytes of the frame contain the target IP address, which is the IP address of the machine that my computer wants to know the MAC address of.
13.

```
Frame 21: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
Ethernet II, Src: Cisco_77:e4:00 (00:25:46:77:e4:00), Dst: HonHaiPr_59:00:19 (34:68:95:59:00:19)
      Destination: HonHaiPr_59:00:19 (34:68:95:59:00:19)
      Source: Cisco_77:e4:00 (00:25:46:77:e4:00)
      Type: ARP (0x0806)
      Padding: 000000000000000000000000000000000000
Address Resolution Protocol (reply)
      Hardware type: Ethernet (1)
      Protocol type: IPv4 (0x0800)
      Hardware size: 6
      Protocol size: 4
      Opcode: reply (2)
      Sender MAC address: Cisco_77:e4:00 (00:25:46:77:e4:00)
      Sender IP address: 10.200.204.1
      Target MAC address: HonHaiPr_59:00:19 (34:68:95:59:00:19)
      Target IP address: 10.200.205.210
```

    a. The ARP opcode begins 7 bytes after the end of the ethernet frame.
    b. The value of the opcode field is `0x0002`, indicating a reply message.
    c. Now the sender MAC and IP address fields fully qualify the machine that the original request had been asking about. In this case, the Sender IP Address matches the previous request's Target IP Address. But now the Sender MAC Address answers the request's blank Target MAC Address field.
14. The source address in the ethernet frame is `00:25:46:77:e4:00`, while the destination address is `34:68:95:59:00:19` (the ethernet address of my computer).

15. The other ARP packet in the example capture was from another computer on the network that was blindly broadcasting its request to the network. Since the request wasn’t relevant to the computer making the capture, it did not have to send a response packet. Additionally, the computer that does respond to the packet has the sender IP from the request, and therefore does not have to send the response as a broadcast message. Because of this, that packet also wouldn’t show up in the example trace.