Before starting to capture the packets, use the command “sudo arp -d” to flush the arp cache. Besides, ipconfig/flushdns must be used to flush the DNS cache.

An ICMP packet broadcasting the address of the router is received.

```
1 0.000000   fe80::d0e1:9aff:f… ff02::1          ICMPv6   174 Router Advertisement from fc:51:a4:16:bf:64
▷ Frame 1: 174 bytes on wire (1392 bits), 174 bytes captured (1392 bits) on interface 0
◢ Ethernet II, Src: ArrisGro_16:bf:64 (fc:51:a4:16:bf:64), Dst: IPv6mcast_01 (33:33:00:00:00:01)
  ▷ Destination: IPv6mcast_01 (33:33:00:00:00:01)
  ▷ Source: ArrisGro_16:bf:64 (fc:51:a4:16:bf:64)
    Type: IPv6 (0x86dd)
▷ Internet Protocol Version 6, Src: fe80::d0e1:9aff:feb7:2705, Dst: ff02::1
▷ Internet Control Message Protocol v6
```

ICMP packet

The source is ArrisGro_16:bf:64, and destination is a multicast interface IPv6mcast_01. In IP packet, this source is replaced by fe80::d0e1:9aff:feb7:2705 (shown in the figure below), which means the default gateway of the network. Host discovers this broadcast message and use the information given to report errors when network problems occur. The destination is replaced by ff02::1, which means all the hosts open for IPv6 communication. ICMP works at networking layer.

```
   连接特定的 DNS 后缀 . . . . . : hsd1.ma.comcast.net
   IPv6 地址 . . . . . . . . . . . : 2601:197:800:dcc2:2815:6bba:ca00:9b92
   临时 IPv6 地址. . . . . . . . . : 2601:197:800:dcc2:2c04:cc6f:ee37:30e4
   本地链接 IPv6 地址. . . . . . . : fe80::2815:6bba:ca00:9b92%11
   IPv4 地址 . . . . . . . . . . . : 10.0.0.125
   子网掩码 . . . . . . . . . . . : 255.255.255.0
   默认网关. . . . . . . . . . . . : fe80::d0e1:9aff:feb7:2705%11
                                     10.0.0.1
```

ipconfig

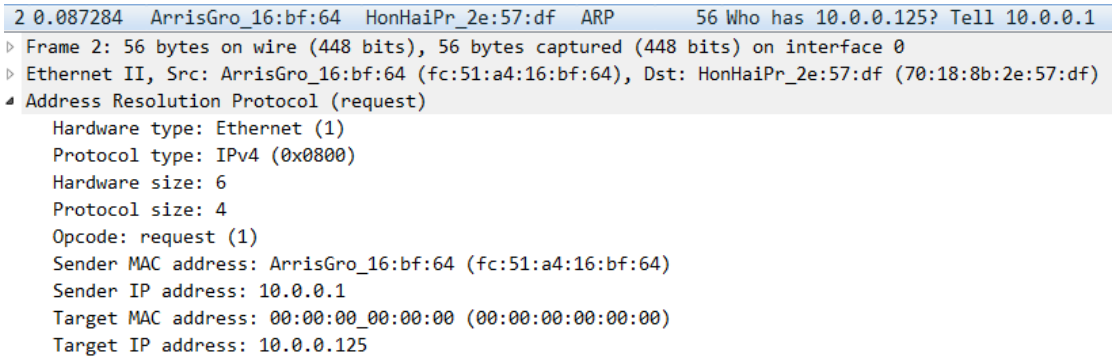
Then the DHCP informs and confirms the IP address of our host.

```
10 6.536362   10.0.0.125     255.255.255.255   DHCP   342 DHCP Inform   - Transaction ID 0xe5028e8c
11 6.544203   10.0.0.1       10.0.0.125        DHCP   342 DHCP ACK      - Transaction ID 0xe5028e8c

▶ Frame 10: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits)
▶ Ethernet II, Src: 70:18:8b:2e:57:df (70:18:8b:2e:57:df), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▶ Internet Protocol Version 4, Src: 10.0.0.125 (10.0.0.125), Dst: 255.255.255.255 (255.255.255.255)
▼ User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)
    Source port: bootpc (68)
    Destination port: bootps (67)
    Length: 308
  ▶ Checksum: 0x670e [validation disabled]
▶ Bootstrap Protocol

▸ Frame 11: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits)
▸ Ethernet II, Src: fc:51:a4:16:bf:64 (fc:51:a4:16:bf:64), Dst: 70:18:8b:2e:57:df (70:18:8b:2e:57:df)
▸ Internet Protocol Version 4, Src: 10.0.0.1 (10.0.0.1), Dst: 10.0.0.125 (10.0.0.125)
▾ User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)
    Source port: bootps (67)
    Destination port: bootpc (68)
    Length: 308
  ▸ Checksum: 0x20c6 [validation disabled]
▸ Bootstrap Protocol
```

DHCP Inform and ACK

The default gateway broadcast message to give the IP address to the our host. Our host gives back a confirm message. DHCP is working on port 67 on gateway and 68 on client, which is defined in RFC. DHCP is based on UDP, working at application layer.

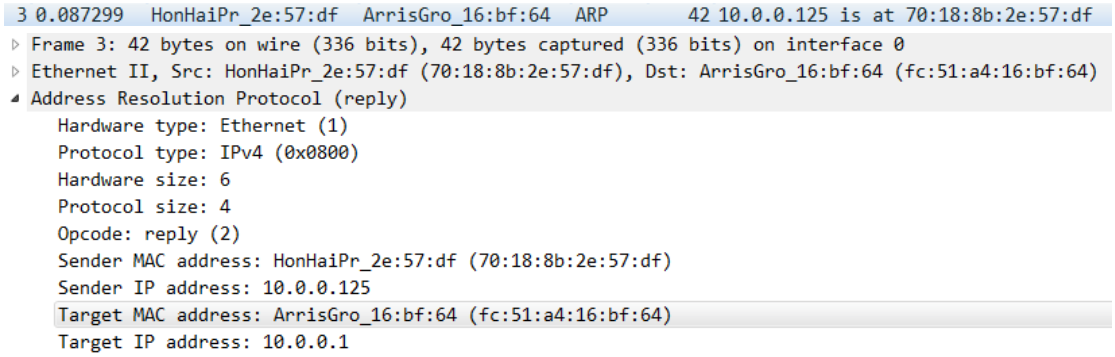
The router needs to know the host's MAC address to deliver the packets. So it broadcasts the message below to find our host 10.0.0.125.



ARP packet

The sender MAC address is ArrisGro_16:bf:64(same as the ICMP packet sender), and its IP address is 10.0.0.1. This is a broadcast message to find the MAC address for the target IP address.

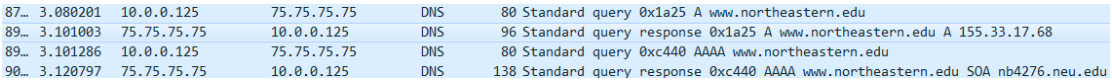
After our host (MAC address HonHaiPr_2e:57:df) received this broadcast message, it respond to the default gateway with its MAC address, like saying "I'm here".



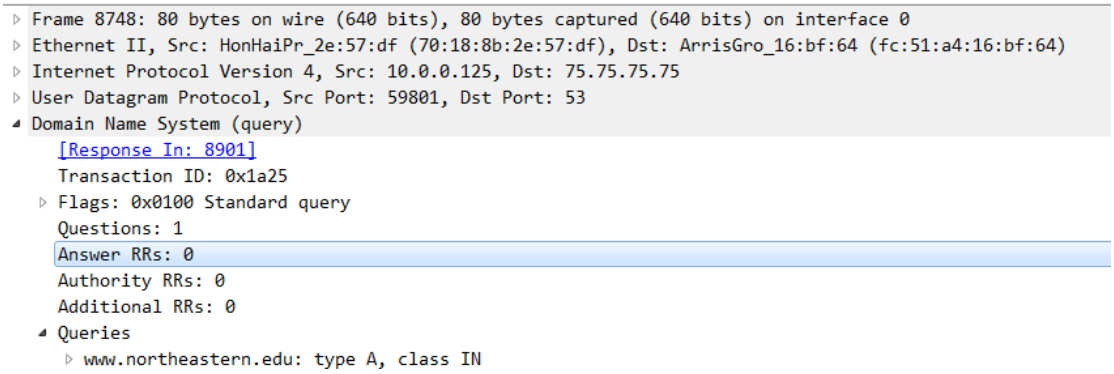
Responding ARP packet

After sending and receiving the ARP packet, both the host and default gateway has cached the information to go on with further information. ARP works at data link layer

Then, the link http://www.northeastern.edu is entered into google chrome. The explorer tries to figure out the IP address of www.northeastern.edu by sending a DNS request.

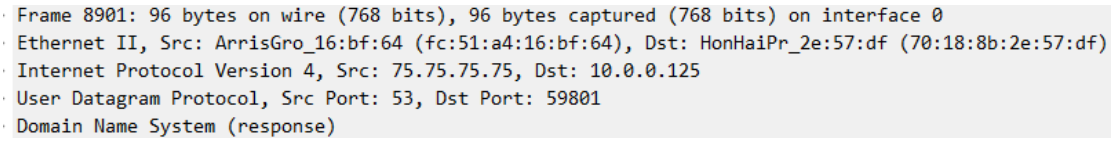


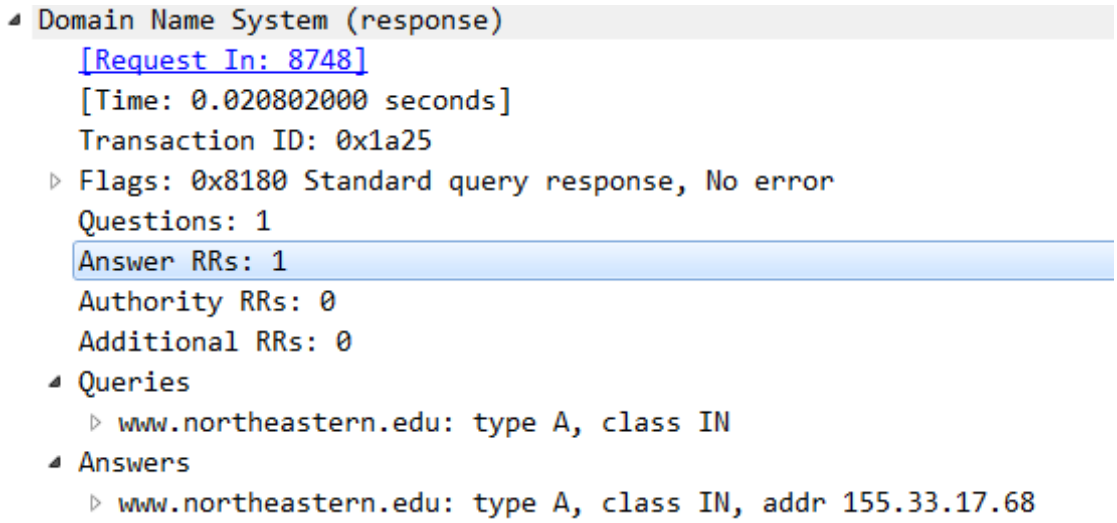
DNS packets



First DNS request

DNS is based on UDP protocol. Our host 10.0.0.125 sends a request for www.northeastern.edu from port 59801 to dns server 75.75.75.75, port 53. After receiving this request packet, DNS server 75.75.75.75 sends the answer back.

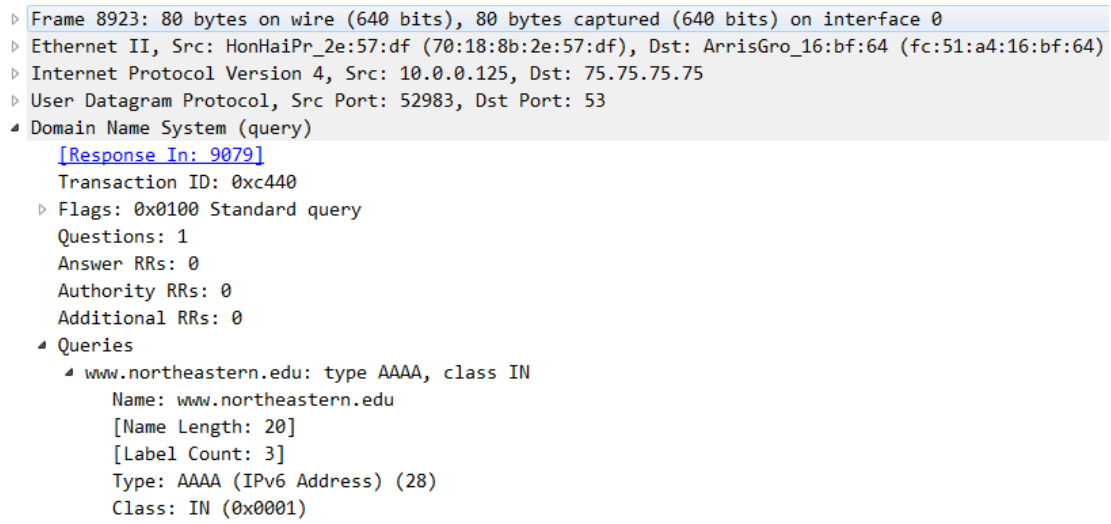




Answer to First DNS request

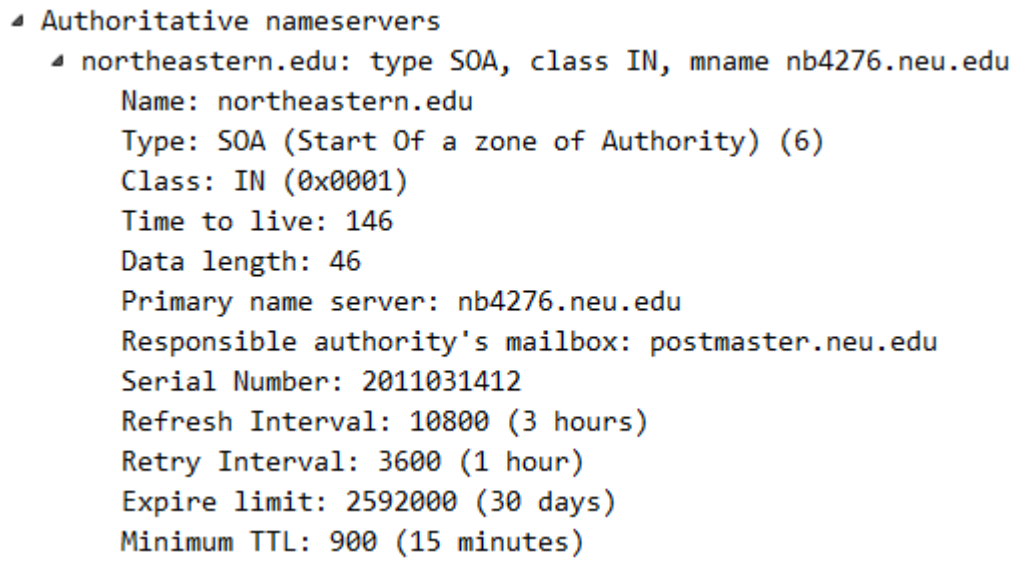
The source IP is 75.75.75.75 source port 53. The destination IP is 10.0.0.125 port 59801. These reversed to those of the request packet. The answer 155.33.17.68 is in the answer part of the DNS answering packet. This is the IP for www.northeastern.edu . There is one answer to one question.

The second query is asking for IPv6 (TYPE AAAA) address for Northeastern.



Second AAAA DNS request

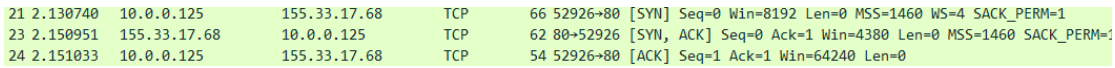
The source and destination IP are the same with the first request, but the port for 10.0.0.125 changes to 52983 and port for 75.75.75.75 remains the same.



Second AAAA DNS answer

The port for client and DNS server remains the same as the request. The answering part for AAAA is different from type A. Instead of an IPv4 address, the DNS server is returning attributes such as name, type, class, TTL, Data Length, Primary name server and so on (shown in the figure above). DNS protocol is located at application layer.

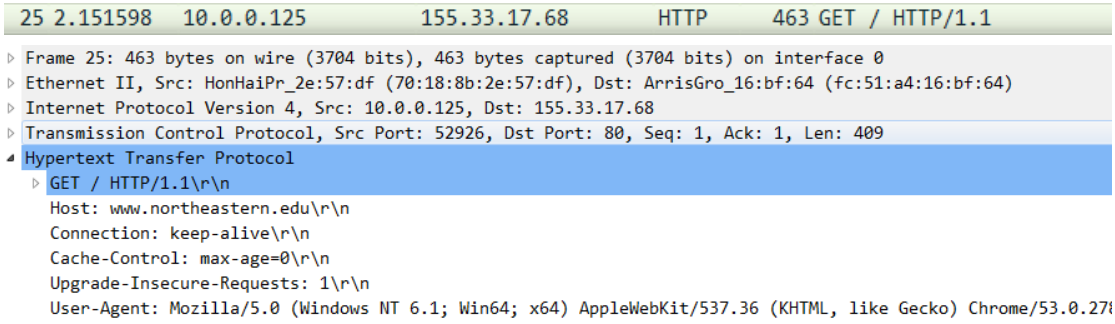
After getting the IP address of http://www.northeastern.edu, the host starts the three way hand shake with www.northeastern.edu . The IP address of www.northeastern.edu is 155.33.17.68 and the port used by server is 80 (HTTP required). The port used by local host or client is random. In this assignment, the port for local host is 52926.



TCP three-way handshake

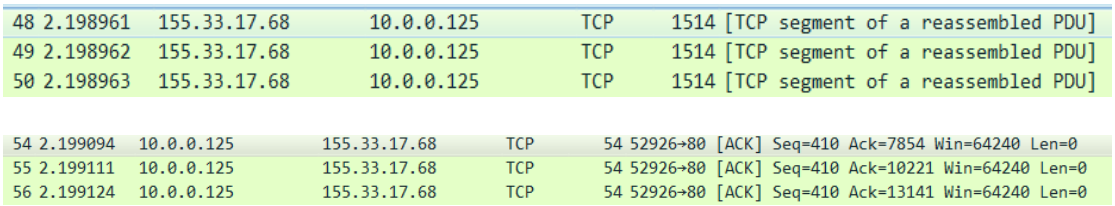
The IP address of local host is 10.0.0.125. The local host sends a TCP packet with Seq = 0 to start the three way handshake. The server receives the first handshake message and returns the second handshake with Seq = 0 and ACK = 1 to confirm the first handshake message. Then the local host receives the second handshake message and returns the third handshake with Seq = 1 and Ack = 1 to confirm the second handshake. The TCP connection is established between the local host and the server.

With the TCP connection established, the local host starts the HTTP by sending an HTTP GET packet to get the webpage requested.



HTTP GET packet

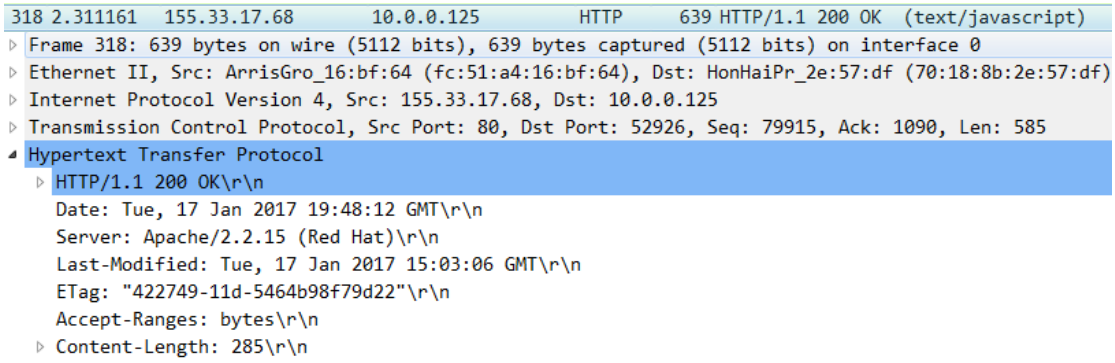
After receiving the HTTP get packet from local host, the server starts to send the requested page to the local host. The server transmits the webpage by TCP segments, and the local host returns the Ack message correspondingly to these segment packets. If there are loss of packets, the server will start the fast retransmission or retransmission until all necessary parts are all delivered. Following is the sample data and Ack message during the connection.



Sample data and Acks

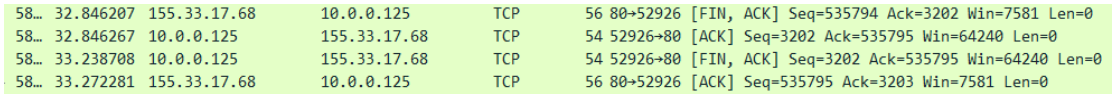
Then, the server sends a 200 OK packet as the answer of the HTTP GET message for each HTTP GET message. Each HTTP GET packet would have a response. Following is a sample of HTTP response

made by the server.



Sample HTTP 200 response

By the end of the whole connection, the server sends a FIN packet as the ending signal of the whole process. The following figure is the ending process of this connection.



Ending process of connection

As shown in the figure given above, the server first send a FIN packet to show that this is the end of transmission. PSH flag can be set in this ending packet to show that there is still some payload in this packet. After receiving this FIN packet, the client or local host send the ACK packet for this FIN packet and another FIN ACK packet to end this connection. Finally, the server returns the final ACK for the last FIN ACK sent by the client. The whole process is ended. Following is the figure of part of flow for this connection. HTTP is based on TCP. It is located at application layer.