

Networking with Linux Lab

Module 4: Analyzing Network traffic

Assignment 12: Monitoring the Network using WireShark

1. Explain how to create a .pcap file in ns3.

Hint: Utilize the `EnablePcapAll()` method and refer to the `first.cc`, `second.cc`, and `third.cc` files.

Aim: To capture and analyze network traffic in ns3 simulations

Theory:

Creating a .pcap file in NS-3 involves capturing network traffic generated during a simulation and saving it in the .pcap format, which is commonly used for packet capture. NS-3 provides a simple way to enable .pcap tracing for all nodes in a network using the `EnablePcapAll()` method. This method is typically called on the point-to-point devices or `NetDevices`.

Here's a step-by-step guide to creating a .pcap file in NS-3:

1. **Include Necessary Headers:** First, include the necessary headers in your simulation script. You'll typically need headers for the network devices, .pcap tracing, and any other modules you're using.
2. **Create the Network Topology:** Define the network topology you want to simulate. This includes creating nodes, setting up point-to-point or other types of connections between them, and configuring their attributes.
3. **Enable Pcap Tracing:** After setting up the network topology, enable pcap tracing for all the nodes in the network. This is done using the `EnablePcapAll()` method, which is typically called on the point-to-point devices or `NetDevices`.
4. **Run the Simulation:** Once you've set up the network and enabled pcap tracing, run the simulation script. During the simulation, NS-3 will capture the network traffic and save it to a .pcap file.
5. **Analyze the .pcap File:** After the simulation has completed, you can analyze the generated .pcap file using tools like Wireshark. This allows you to inspect the captured packets, analyze network behavior, and debug your simulation if necessary.

Code:

> **first.cc**

```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/applications-module.h"
#include "ns3/mobility-helper.h"
#include "ns3/netanim-module.h"

// Default Network Topology
//
//      10.1.1.0
// n0 ----- n1
//  point-to-point
//

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("FirstScriptExample");

int
main (int argc, char *argv[])
{
    CommandLine cmd (__FILE__);
    cmd.Parse (argc, argv);

    Time::SetResolution (Time::NS);
    LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
    LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);

    NodeContainer nodes;
    nodes.Create (2);

    PointToPointHelper pointToPoint;
    pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
    pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));

    NetDeviceContainer devices;
    devices = pointToPoint.Install (nodes);

    InternetStackHelper stack;
    stack.Install (nodes);

    Ipv4AddressHelper address;
```

```
address.SetBase ("10.1.1.0", "255.255.255.0");

Ipv4InterfaceContainer interfaces = address.Assign (devices);

UdpEchoServerHelper echoServer (9);

ApplicationContainer serverApps = echoServer.Install (nodes.Get (1));
serverApps.Start (Seconds (1.0));
serverApps.Stop (Seconds (10.0));

UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);
echoClient.SetAttribute ("MaxPackets", UIntegerValue (1));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute ("PacketSize", UIntegerValue (1024));

ApplicationContainer clientApps = echoClient.Install (nodes.Get (0));
clientApps.Start (Seconds (2.0));
clientApps.Stop (Seconds (10.0));

// NetAnimation ---before simulator run
MobilityHelper mobility;
mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
mobility.Install(nodes);

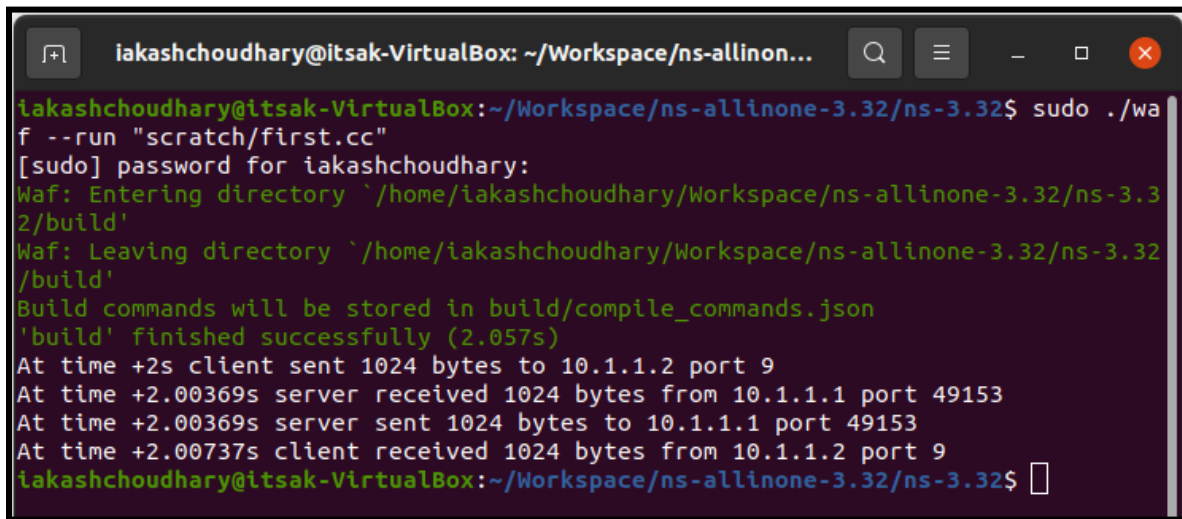
AnimationInterface anim("first.xml");
AnimationInterface::SetConstantPosition(nodes.Get(0), 10, 25);
AnimationInterface::SetConstantPosition(nodes.Get(1), 40, 25);
anim.EnablePacketMetadata(true);

// Creating .pcap files for Wireshark-pcap: packet capture information
pointToPoint.EnablePcapAll("first");

Simulator::Run ();
Simulator::Destroy ();
return 0;
}
```

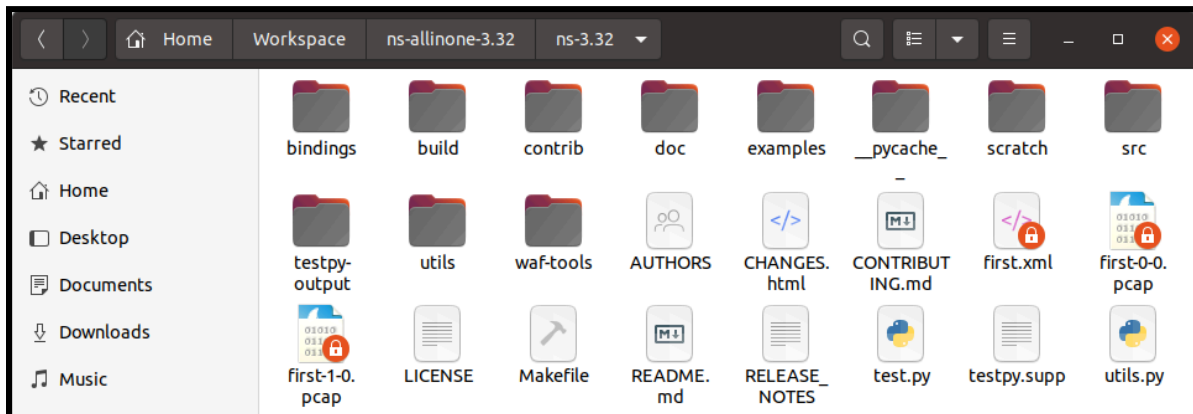
Command & Screenshot:

> `$ sudo ./waf --run "scratch/first.cc"`



```
iakashchoudhary@itsak-VirtualBox: ~/Workspace/ns-allinon...  
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$ sudo ./waf --run "scratch/first.cc"  
[sudo] password for iakashchoudhary:  
Waf: Entering directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'  
Waf: Leaving directory `/home/iakashchoudhary/Workspace/ns-allinone-3.32/ns-3.32/build'  
Build commands will be stored in build/compile_commands.json  
'build' finished successfully (2.057s)  
At time +2s client sent 1024 bytes to 10.1.1.2 port 9  
At time +2.00369s server received 1024 bytes from 10.1.1.1 port 49153  
At time +2.00369s server sent 1024 bytes to 10.1.1.1 port 49153  
At time +2.00737s client received 1024 bytes from 10.1.1.2 port 9  
iakashchoudhary@itsak-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$
```

> After **successfully executing** the above **command**, a **.pcap** file will be **created**.



Conclusion: Enabling .pcap functionality in ns3 facilitates packet capture and analysis for network simulations.

2. Analyze the network traffic captured in the first and third (.pcap) files.

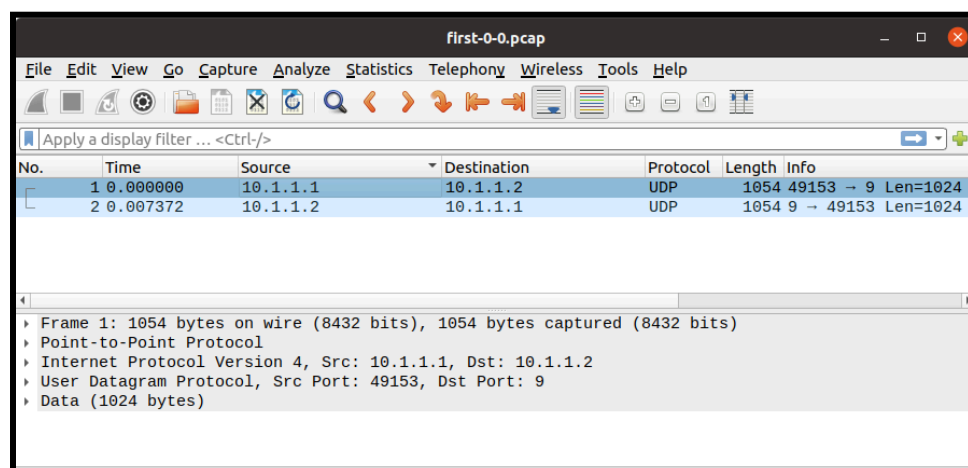
Aim: To understand and evaluate the communication patterns, potential security threats, and overall network performance

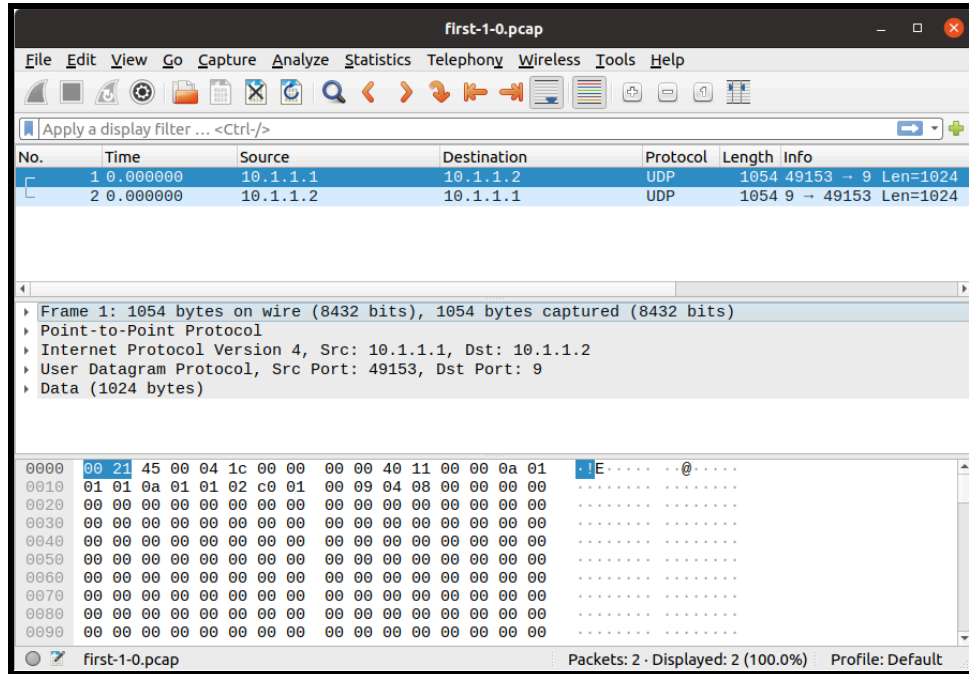
Theory:

Analyzing network traffic captured in .pcap files involves several key steps and theories:

1. **Packet Inspection:** Examine individual packets within the .pcap files to understand their structure, headers, and payload contents.
2. **Protocol Analysis:** Identify the protocols used in the communication (e.g., HTTP, TCP, UDP) to understand how data is being transmitted.
3. **Traffic Patterns:** Analyze the patterns of communication, such as frequency of requests, data volume transferred, and timing of interactions.
4. **Anomaly Detection:** Look for any irregularities or suspicious activity in the network traffic that could indicate security threats or performance issues.
5. **Traffic Flow Analysis:** Determine the flow of traffic between network nodes, including source and destination addresses, ports, and protocols.
6. **Performance Metrics:** Measure network performance metrics such as latency, throughput, and packet loss to assess the efficiency of communication.
7. **Interpretation:** Interpret the findings to draw conclusions about the behavior of the network, potential issues, and areas for optimization or security enhancement.

Screenshot:





2.1 Steps to capture live data packets.

Theory:

1. **Choose a Tool:** Select a packet capturing tool like Wireshark or tcpdump.
2. **Launch the Tool:** Open the chosen tool on your device.
3. **Select Interface:** Choose the network interface you want to monitor (e.g., Wi-Fi, Ethernet).
4. **Start Capturing:** Begin capturing packets by clicking the "Start" or "Capture" button.
5. **Analyze Data:** Review captured packets for relevant information like source, destination, and protocol.
6. **Filter Data (Optional):** Use filters to refine captured data based on specific criteria.
7. **Stop Capturing:** End the packet capture process when you've collected enough data.
8. **Save Data (Optional):** Save captured packets for further analysis or documentation.

2.2. Explain the three sections of Wireshark with examples.

Theory:

1. **Packet List Pane:** This section displays a list of captured packets in a tabular format. Each row represents a single packet, and columns provide details such as the packet number, time of capture, source and destination addresses, protocol type, and packet length. This pane gives you a quick overview of all captured network traffic. Example: In this pane, you might see rows representing different types of packets such as Ethernet frames, IP packets, TCP segments, or UDP datagrams. You can examine each packet to analyze its contents and understand the communication between network hosts.

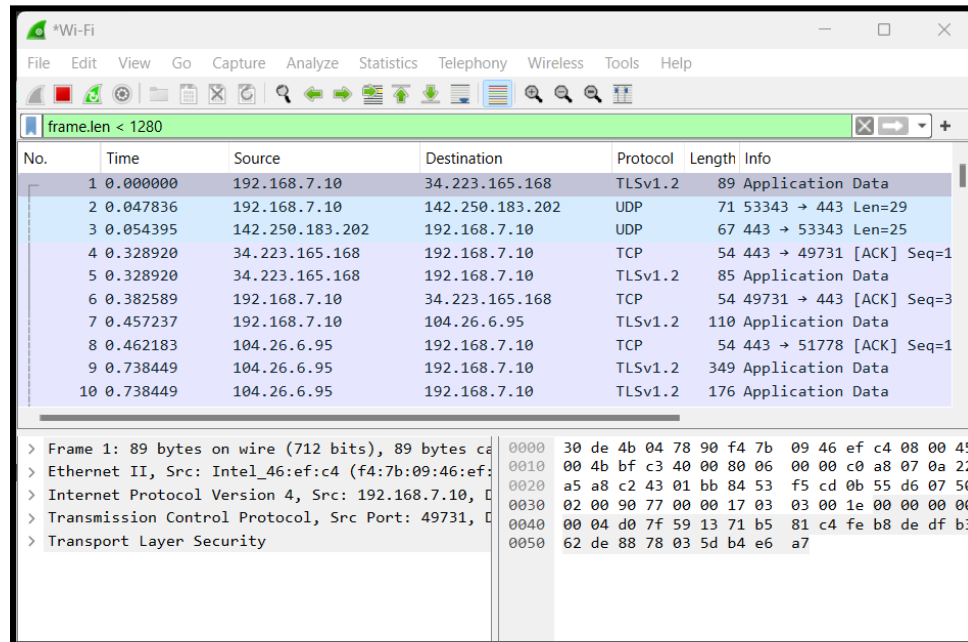
2. **Packet Details Pane:** When you select a packet from the Packet List Pane, its details are displayed in this section. It provides a hierarchical view of the packet contents, showing various protocol layers encapsulated within the packet. Each layer contains fields and their corresponding values, allowing you to inspect the headers and payloads of different network protocols. Example: If you select an HTTP packet, you'll see the Ethernet frame header, followed by the IP header, then the TCP header, and finally the HTTP header and body. You can analyze the HTTP request and response messages, including URLs, headers, and data payloads.

3. **Packet Bytes Pane:** This section provides a hexadecimal and ASCII representation of the selected packet's raw bytes. It allows you to view the packet's binary data in detail, which can be useful for analyzing packet structures, identifying patterns, or troubleshooting issues at the byte level. Example: If you're investigating a network protocol or trying to diagnose a network problem, you can use the Packet Bytes Pane to examine the raw packet contents. You might look for specific byte sequences, protocol signatures, or anomalies in the data.

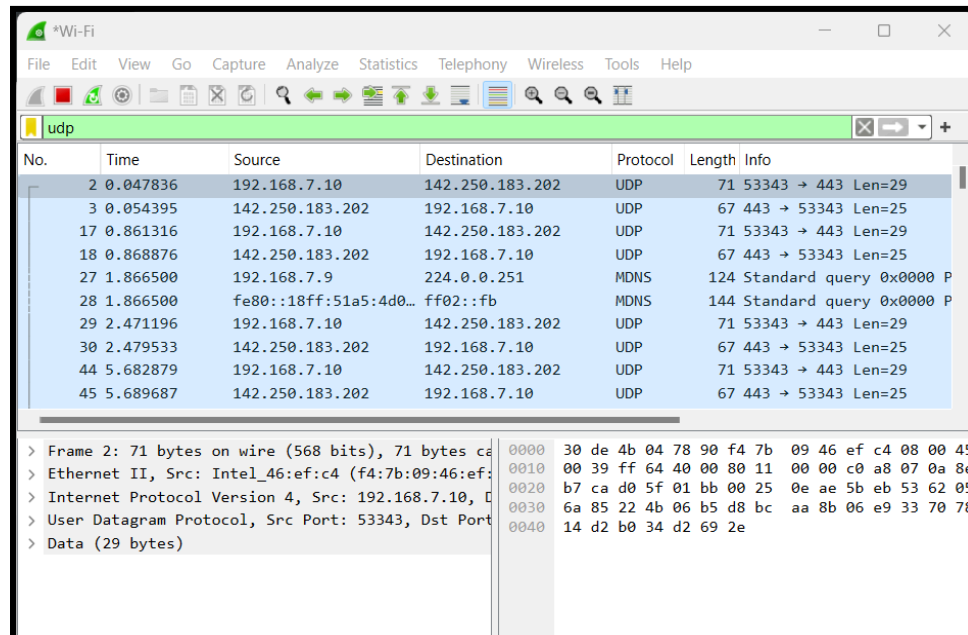
2.3. Perform at least four filter expressions in the Wireshark tool on any packet (live or offline).

Screenshot:

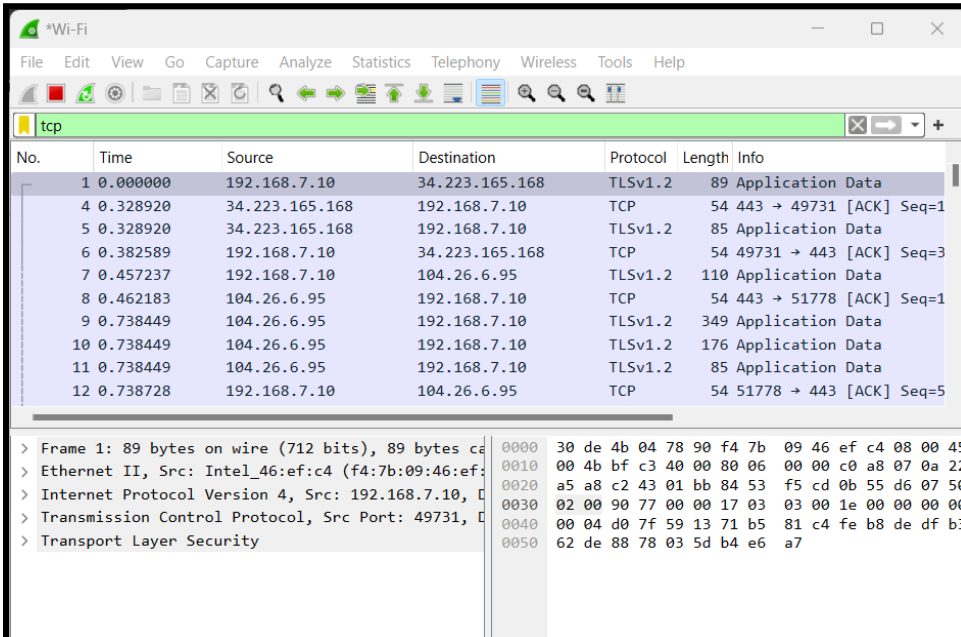
1. frame.len < 1280



2. udp



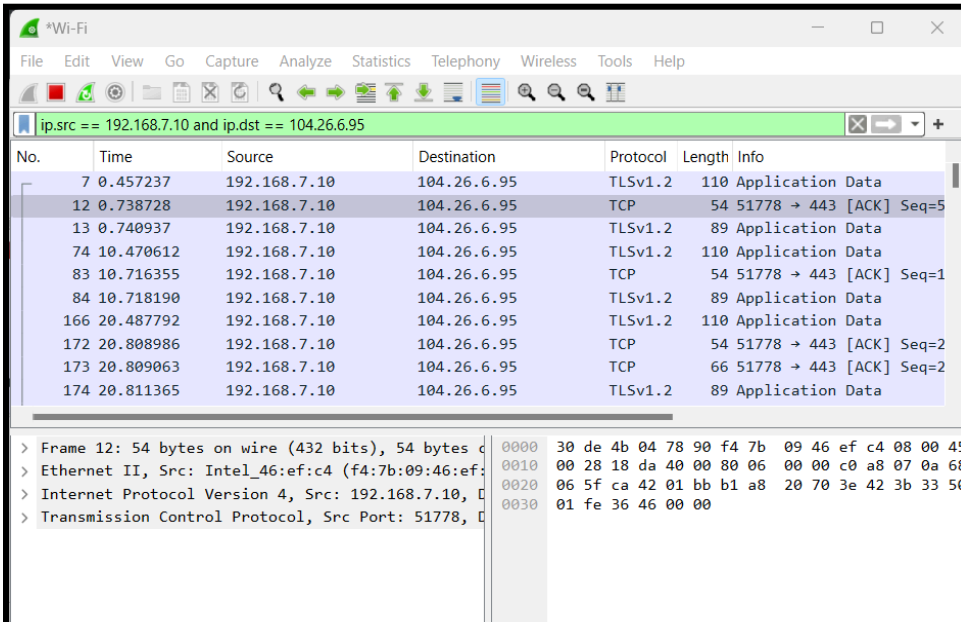
3. tcp



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.7.10	34.223.165.168	TLSv1.2	89	Application Data
4	0.328920	34.223.165.168	192.168.7.10	TCP	54	443 → 49731 [ACK] Seq=1
5	0.328920	34.223.165.168	192.168.7.10	TLSv1.2	85	Application Data
6	0.382589	192.168.7.10	34.223.165.168	TCP	54	49731 → 443 [ACK] Seq=3
7	0.457237	192.168.7.10	104.26.6.95	TLSv1.2	110	Application Data
8	0.462183	104.26.6.95	192.168.7.10	TCP	54	443 → 51778 [ACK] Seq=1
9	0.738449	104.26.6.95	192.168.7.10	TLSv1.2	349	Application Data
10	0.738449	104.26.6.95	192.168.7.10	TLSv1.2	176	Application Data
11	0.738449	104.26.6.95	192.168.7.10	TLSv1.2	85	Application Data
12	0.738728	192.168.7.10	104.26.6.95	TCP	54	51778 → 443 [ACK] Seq=5

> Frame 1: 89 bytes on wire (712 bits), 89 bytes captured (712 bits) on interface 0
 > Ethernet II, Src: Intel_46:ef:c4 (f4:7b:09:46:ef:c4), Dst: Intel_46:ef:c4 (f4:7b:09:46:ef:c4)
 > Internet Protocol Version 4, Src: 192.168.7.10, Dst: 34.223.165.168
 > Transmission Control Protocol, Src Port: 49731, Dst Port: 443
 > Transport Layer Security

4. ip.src == 192.168.7.10 and ip.dst == 104.26.6.95



No.	Time	Source	Destination	Protocol	Length	Info
7	0.457237	192.168.7.10	104.26.6.95	TLSv1.2	110	Application Data
12	0.738728	192.168.7.10	104.26.6.95	TCP	54	51778 → 443 [ACK] Seq=5
13	0.740937	192.168.7.10	104.26.6.95	TLSv1.2	89	Application Data
74	10.470612	192.168.7.10	104.26.6.95	TLSv1.2	110	Application Data
83	10.716355	192.168.7.10	104.26.6.95	TCP	54	51778 → 443 [ACK] Seq=1
84	10.718190	192.168.7.10	104.26.6.95	TLSv1.2	89	Application Data
166	20.487792	192.168.7.10	104.26.6.95	TLSv1.2	110	Application Data
172	20.808986	192.168.7.10	104.26.6.95	TCP	54	51778 → 443 [ACK] Seq=2
173	20.809063	192.168.7.10	104.26.6.95	TCP	66	51778 → 443 [ACK] Seq=2
174	20.811365	192.168.7.10	104.26.6.95	TLSv1.2	89	Application Data

> Frame 12: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
 > Ethernet II, Src: Intel_46:ef:c4 (f4:7b:09:46:ef:c4), Dst: Intel_46:ef:c4 (f4:7b:09:46:ef:c4)
 > Internet Protocol Version 4, Src: 192.168.7.10, Dst: 104.26.6.95
 > Transmission Control Protocol, Src Port: 51778, Dst Port: 443

2.4. Does Wireshark impact performance?

Theory:

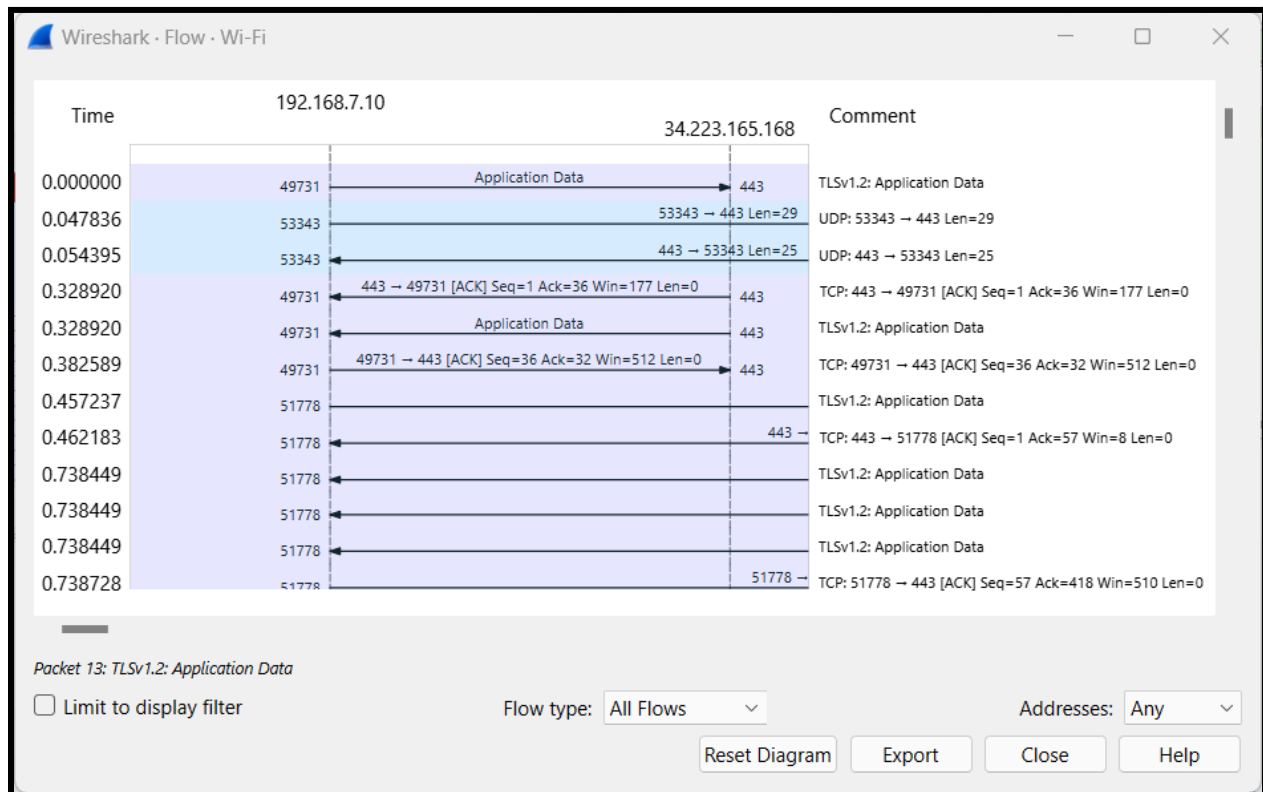
No, Wireshark doesn't impact performance. It typically has minimal impact on performance during normal usage. Its overhead is generally negligible unless you're capturing a large amount of network traffic or running it on a system with very limited resources. However, in most cases, the impact should be insignificant.

2.5. Flowgraph.

Theory:

In Wireshark, a flowgraph is a graphical representation of network traffic patterns over time. It displays the flow of packets between hosts or network devices, helping users visualize communication patterns and identify anomalies. Flowgraphs in Wireshark typically show packet interactions, such as connections, data transfers, and protocol exchanges, which can be useful for troubleshooting network issues, analyzing performance, or investigating security incidents.

Screenshot:

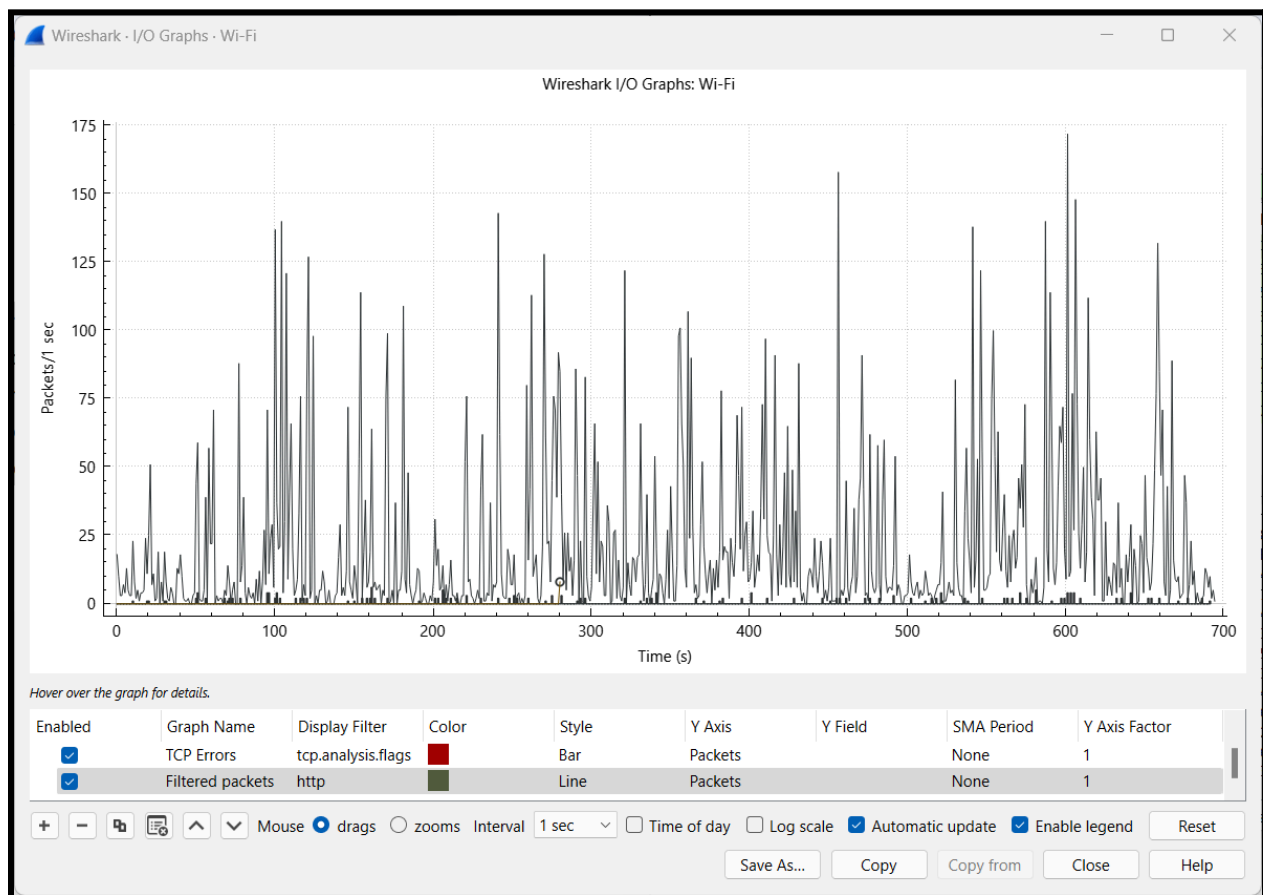


2.6. I/O Graph.

Theory:

An I/O Graph in Wireshark is a graphical representation of input and output (I/O) traffic over time. It provides a visual depiction of network activity, making it easier to identify patterns, trends, and anomalies. It typically plots parameters like packet count, packet size, throughput, or any other metrics captured by Wireshark over a specified time range. This visual representation aids in network analysis and troubleshooting by allowing users to quickly identify network issues or unusual behavior.

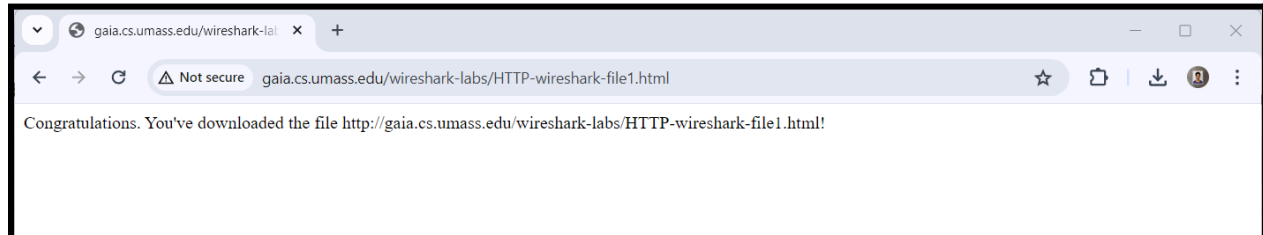
Screenshot:



Conclusion: Analyzing network traffic captured in the provided .pcap files reveals patterns of communication and potential security threats.

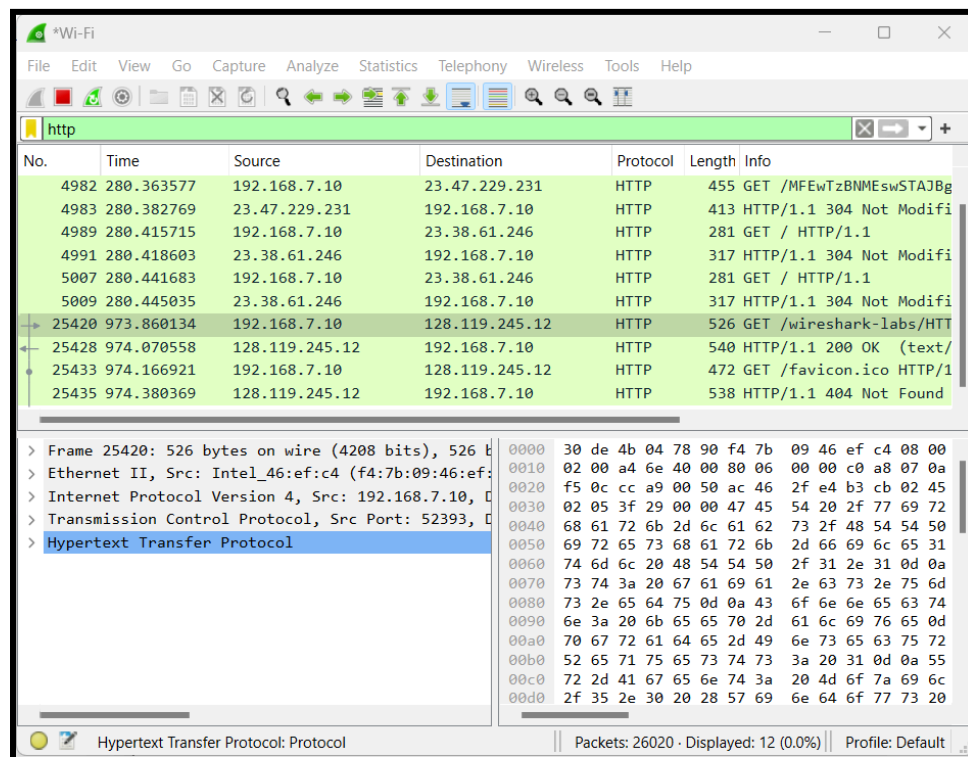
3. Use Wireshark to capture packets by browsing <http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html> to answer the following questions:

Screenshot:



a. Filter the packets using "http."

Screenshot:



b. What version of HTTP is the server running?**Screenshot:**

```

> Transmission Control Protocol, Src Port: 52393, Dst Port: 80, Seq: 1, Ack:
  Hypertext Transfer Protocol
    GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1\r\n
      [Expert Info (Chat/Sequence): GET /wireshark-labs/HTTP-wireshark-file1.html]
      Request Method: GET
      Request URI: /wireshark-labs/HTTP-wireshark-file1.html
      Request Version: HTTP/1.1
      Host: gaia.cs.umass.edu\r\n
  
```

c. What languages (if any) does your browser indicate that it can accept to the server?**Screenshot:**

```

Upgrade-Insecure-Requests: 1\r\n
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif
Accept-Encoding: gzip, deflate\r\n
Accept-Language: en-US,en;q=0.9\r\n
\r\n
[Full request URI: http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html]
[HTTP request 1/2]
  
```

d. What is the IP address of your computer? What is the IP address of the server?**Screenshot:**

```

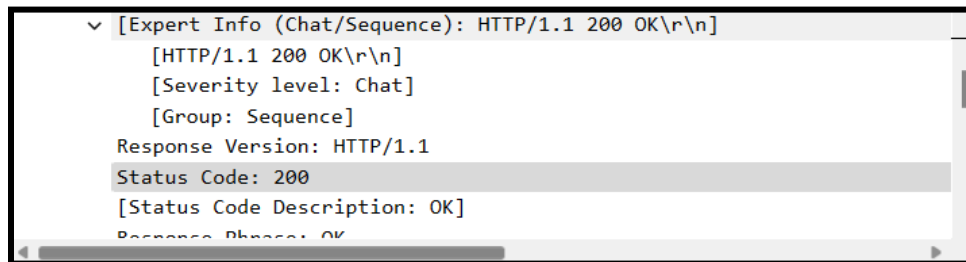
Wireless LAN adapter Wi-Fi:

Connection-specific DNS Suffix . : 
Link-local IPv6 Address . . . . . : fe80::c19c:79de:ff2c:6462%5
IPv4 Address. . . . . : 192.168.7.10
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.7.1
  
```


No.	Time	Source	Destination	Protocol	Length	Info
25420	973.860134	192.168.7.10	128.119.245.12	HTTP	526	GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1
25428	974.070558	128.119.245.12	192.168.7.10	HTTP	540	HTTP/1.1 200 OK (text/html)
25433	974.166921	192.168.7.10	128.119.245.12	HTTP	472	GET /favicon.ico HTTP/1.1
25435	974.380369	128.119.245.12	192.168.7.10	HTTP	538	HTTP/1.1 404 Not Found

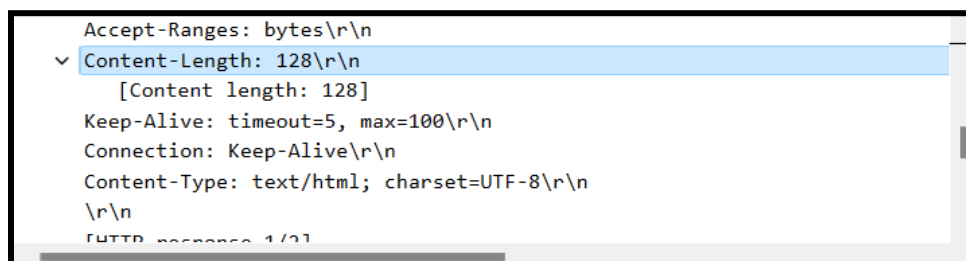
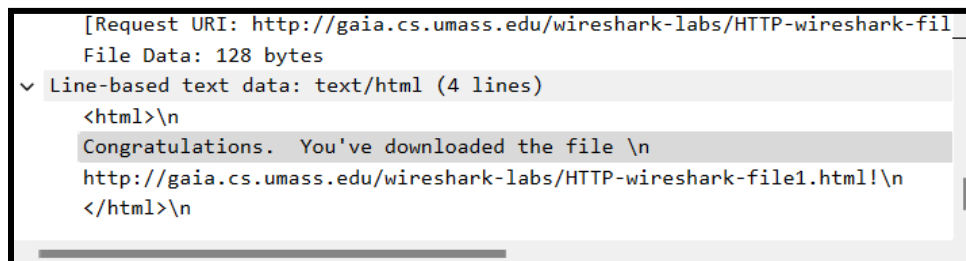
e. What is the status code returned from the server to your browser?

Screenshot:



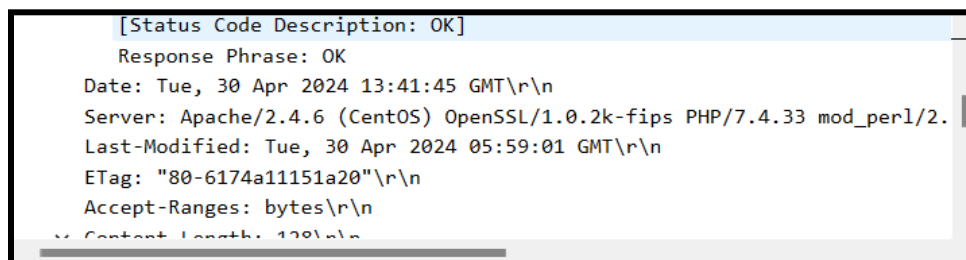
f. Is the server returning any content? If so, how many bytes of content are being returned to your browser?

Screenshot:



g. When was the HTML file that you are retrieving last modified at the server?

Screenshot:



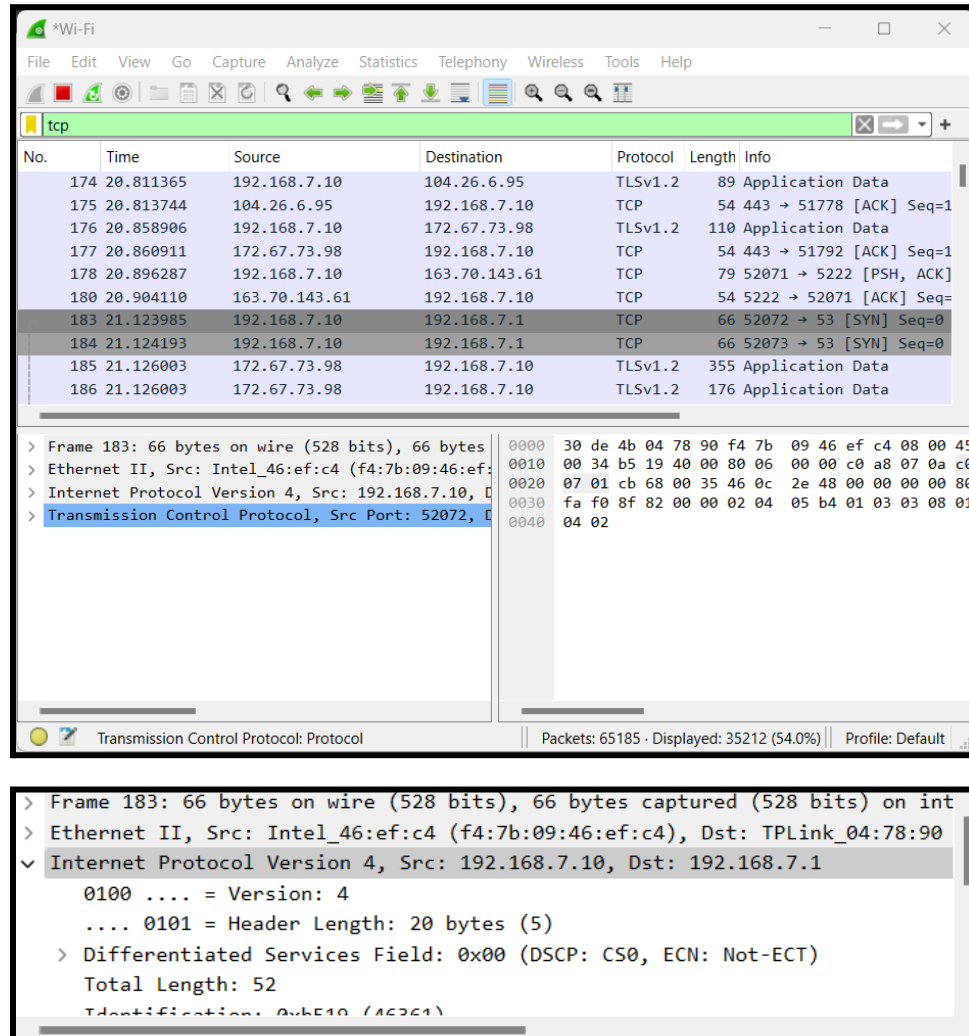
4. Use Wireshark to capture packets by browsing your college website and answer the following questions:

Screenshot:



a. Choose any packet. What are the IP addresses of the source and destination?

Screenshot:



b. What is the value in the Time-to-Live (TTL) field in this IPv4 datagram's header? What does it mean?

Theory:

The TTL field serves multiple purposes:

1. **Preventing loops:** It ensures that packets don't endlessly circulate in a network due to routing errors or other issues.

2. Determining packet freshness: It provides a means to gauge the "freshness" or "age" of a packet, as it decreases with each hop. This is particularly useful for routing protocols to prevent the propagation of outdated routing information.

Screenshot:

```
Total Length: 52
Identification: 0xb519 (46361)
> 010. .... = Flags: 0x2, Don't fragment
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 128
Protocol: TCP (6)
Header Checksum: 0x0000 [validation disabled]
[Header checksum status: Unverified]
```

c. How many bytes are in the IP header?

Screenshot:

```
> Frame 183: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on int
> Ethernet II, Src: Intel_46:ef:c4 (f4:7b:09:46:ef:c4), Dst: TPLink_04:78:90
v Internet Protocol Version 4, Src: 192.168.7.10, Dst: 192.168.7.1
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 52
Identification: 0xb519 (46361)
```

d. How many bytes are in the payload of the IP datagram?

Screenshot:

```
Checksum: 0xe1ce [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> [Timestamps]
> [SEQ/ACK analysis]
TCP payload (301 bytes)
> Transport Layer Security
```

e. Explain how you determined the number of payload bytes.**Theory:**

The TCP header typically consists of 20 bytes, and the payload size can be calculated by subtracting the header length from the total length of the packet.

If the total length of the packet is 301 bytes and the TCP header length is 20 bytes, then the payload size would indeed be 281 bytes.

So, Payload size = Total length of packet - TCP header length

Payload size = 301 bytes - 20 bytes

Payload size = 281 bytes

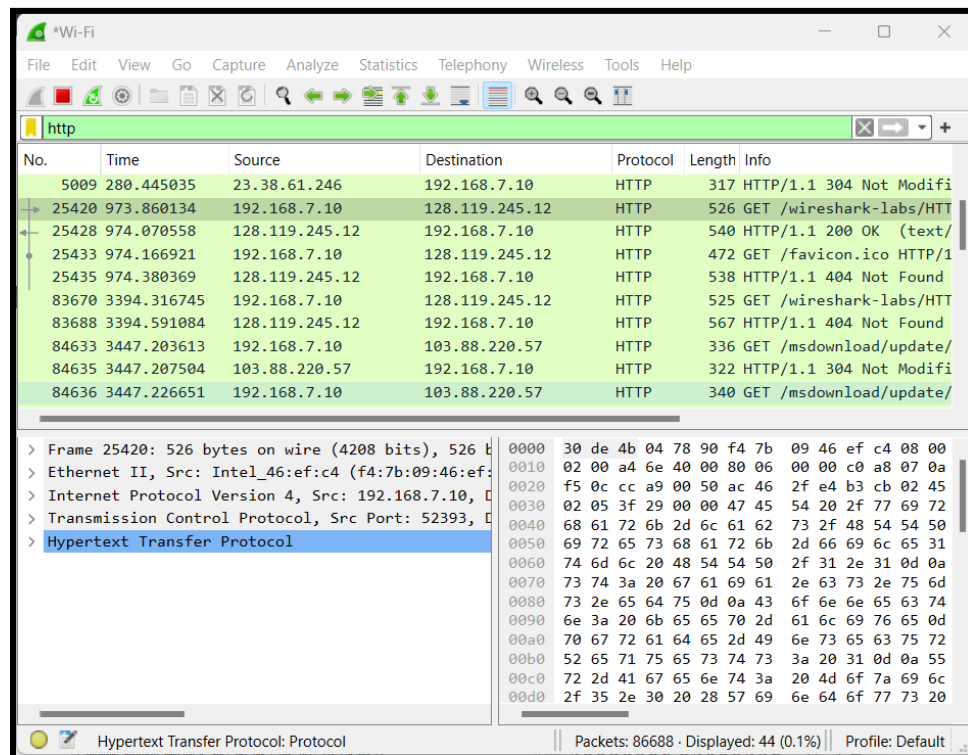
5. Use Wireshark to capture packets by browsing <http://gaia.cs.umass.edu/wireshark-labs/HTTP-wiresharkfile2.html> and answer the following questions:

Screenshot:



a. Filter the packets using "http."

Screenshot:



b. Inspect the contents of the first HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE" line in the HTTP GET?

Theory:

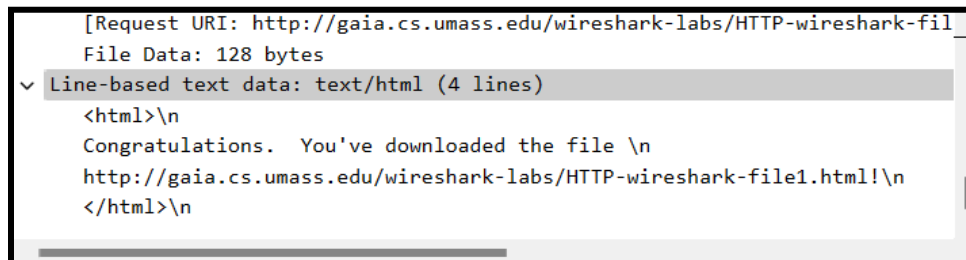
No, there is no IF-MODIFIED-SINCE line in the GET message.

c. Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell?

Theory:

The server did explicitly return the contents of the file. Wireshark includes a section titled "Line-Based Text Data" which shows what the server sent back to my browser which is specifically what the website showed when I brought it up on my browser.

Screenshot:



```
[Request URI: http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html]
File Data: 128 bytes
Line-based text data: text/html (4 lines)
<html>\n
Congratulations. You've downloaded the file \n
http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html!\n
</html>\n
```

d. Now inspect the contents of the second HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE:" line in the HTTP GET? If so, what information follows the "IF-MODIFIED-SINCE:" header?

Theory:

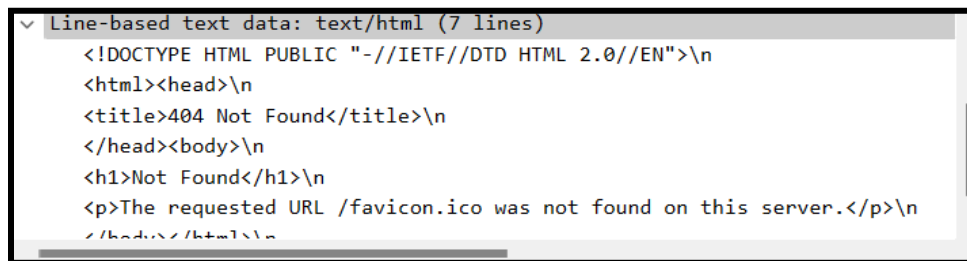
No, there is no IF-MODIFIED-SINCE line in the GET message.

e. What is the HTTP status code and phrase returned from the server in response to this second HTTP GET? Did the server explicitly return the contents of the file? Explain.

Theory:

The HTTP status code is “404: Not Modified”. The server did not return the contents of the file because the browser simply retrieved the contents from its cache. Had the file been modified since it was last accessed, it would have returned the contents of the file, instead it simply told my browser to retrieve the old file from its cached memory.

Screenshot:

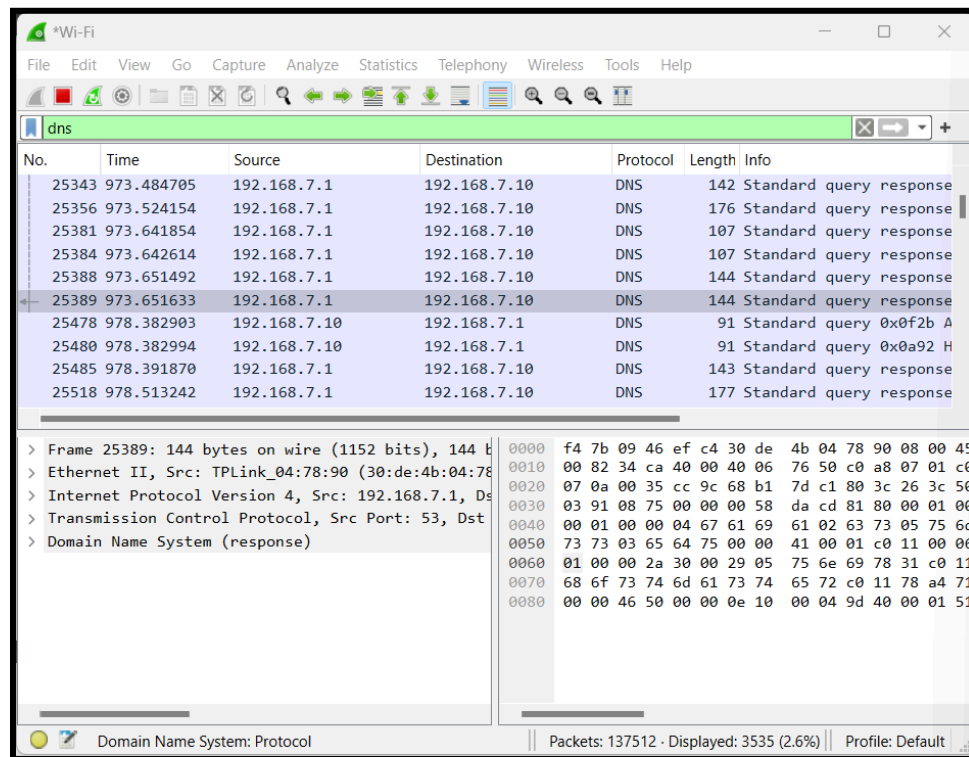
A screenshot of a web browser's developer console. The console shows a message: "Line-based text data: text/html (7 lines)". Below this, the HTML response is displayed as preformatted text. The HTML starts with a DOCTYPE declaration, followed by a head section containing a title "404 Not Found". The body section contains an h1 tag with the text "Not Found" and a paragraph stating "The requested URL /favicon.ico was not found on this server.".

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">\n<html><head>\n<title>404 Not Found</title>\n</head><body>\n<h1>Not Found</h1>\n<p>The requested URL /favicon.ico was not found on this server.</p>\n</body></html>
```

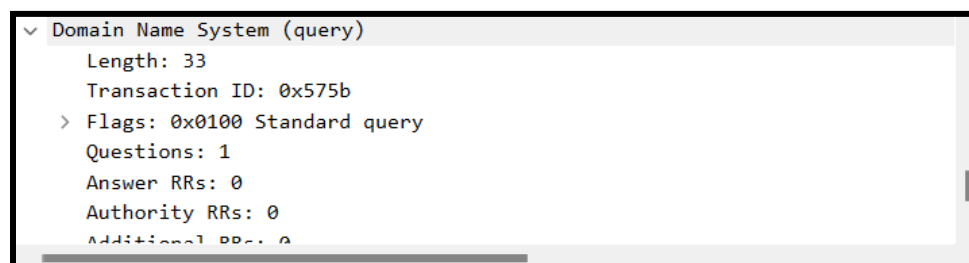
6. Use Wireshark to capture packets by browsing your college website and answer the following questions:

Screenshot:



a. Filter the packets using "DNS."**Screenshot:**

b. Locate the first DNS query message. What is the packet number in the trace for the DNS query message? Is this query message sent over UDP or TCP?

Screenshot:


```

> Frame 197: 87 bytes on wire (696 bits), 87 bytes captured (696 bits) on int
> Ethernet II, Src: Intel_46:ef:c4 (f4:7b:09:46:ef:c4), Dst: TPLink_04:78:90
> Internet Protocol Version 4, Src: 192.168.7.10, Dst: 192.168.7.1
✓ Transmission Control Protocol, Src Port: 52073, Dst Port: 53, Seq: 3, Ack:
  Source Port: 52073
  Destination Port: 53
  [Stream index: 19]
  [Conversation completeness: Complete WITH DATA (31)]

```

c. Now locate the corresponding DNS response to the initial DNS query. What is the packet number in the trace for the DNS response message? Is this response message received via UDP or TCP?

Screenshot:

```

> Transmission Control Protocol, Src Port: 53, Dst Port: 52073, Seq: 1, Ack:
✓ Domain Name System (response)
  Length: 93
  Transaction ID: 0x575b
  > Flags: 0x8180 Standard query response, No error
  Questions: 1
  Answer RRs: 0
  Authority RRs: 1

```

d. What is the destination port for the DNS query message? What is the source port of the DNS response message?

Screenshot:

```

Header Checksum: 0x0000 [validation disabled]
[Header checksum status: Unverified]
Source Address: 192.168.7.10
Destination Address: 192.168.7.1
> Transmission Control Protocol, Src Port: 52073, Dst Port: 53, Seq: 3, Ack:
> [2 Reassembled TCP Segments (35 bytes): #196(2), #197(33)]
> Domain Name System (query)

```

```

Protocol: TCP (6)
Header Checksum: 0x3496 [validation disabled]
[Header checksum status: Unverified]
Source Address: 192.168.7.1
Destination Address: 192.168.7.10
> Transmission Control Protocol, Src Port: 53, Dst Port: 52073, Seq: 1, Ack:
> Domain Name System (response)

```

e. To what IP address is the DNS query message sent?

Screenshot:

```
Wireless LAN adapter Wi-Fi:
    Connection-specific DNS Suffix . : 
    Link-local IPv6 Address . . . . . : fe80::c19c:79de:ff2c:6462%5
    IPv4 Address. . . . . : 192.168.7.10
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.7.1
```

7. Download the dhcp.pcap file from <https://www.cloudshark.org/captures/19585c567c37> (latest) and use Wireshark to answer the following questions:

Screenshot:

The screenshot shows the CloudShark web interface for a capture file named 'dhcp-homenetwork.pcap'. The interface displays a list of 12 packets. The first packet (No. 1) is selected, showing details for a DHCP Inform message. The packet details include Ethernet II, Internet Protocol Version 4, and Dynamic Host Configuration Protocol (Inform). The packet bytes are displayed in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.111	255.255.255.255	DHCP	342	DHCP Inform - Transaction ID 0x9959f928
2	0.006966	192.168.1.1	192.168.1.111	DHCP	590	DHCP ACK - Transaction ID 0x9959f928
3	34.107397	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Transaction ID 0x99314172
4	101.951644	192.168.1.111	255.255.255.255	DHCP	342	DHCP Inform - Transaction ID 0x8b6e2f0
5	101.955851	192.168.1.1	192.168.1.111	DHCP	590	DHCP ACK - Transaction ID 0x8b6e2f0
6	138.244056	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Transaction ID 0x99314174
7	217.544989	192.168.1.111	255.255.255.255	DHCP	342	DHCP Inform - Transaction ID 0x8dd81719
8	217.548858	192.168.1.1	192.168.1.111	DHCP	590	DHCP ACK - Transaction ID 0x8dd81719
9	346.684708	192.168.1.111	255.255.255.255	DHCP	342	DHCP Inform - Transaction ID 0x57ac92de
10	346.688406	192.168.1.1	192.168.1.111	DHCP	590	DHCP ACK - Transaction ID 0x57ac92de
11	430.352810	0.0.0.0	255.255.255.255	DHCP	356	DHCP Request - Transaction ID 0xe01cc718
12	430.428134	192.168.1.1	255.255.255.255	DHCP	590	DHCP NAK - Transaction ID 0xe01cc718

Frame 1: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on
 Ethernet II, Src: Apple_dc:8d:76 (28:cf:da:dc:8d:76), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
 Internet Protocol Version 4, Src: 192.168.1.111, Dst: 255.255.255.255
 User Datagram Protocol, Src Port: 68, Dst Port: 67
 Dynamic Host Configuration Protocol (Inform)

a. Filter the packets using "dhcp." Let's start by looking at the DHCP Discover message. Locate the IP datagram containing the first Discover message in your trace.

Screenshot:

dhcpc-homenetwork.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

dhcp

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.111	255.255.255.255	DHCP	342	DHCP Inform - Tr
2	0.006966	192.168.1.1	192.168.1.111	DHCP	590	DHCP ACK - Tr
3	34.107397	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Tr
4	101.951644	192.168.1.111	255.255.255.255	DHCP	342	DHCP Inform - Tr
5	101.955851	192.168.1.1	192.168.1.111	DHCP	590	DHCP ACK - Tr
6	138.244056	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Tr
7	217.544989	192.168.1.111	255.255.255.255	DHCP	342	DHCP Inform - Tr
8	217.548858	192.168.1.1	192.168.1.111	DHCP	590	DHCP ACK - Tr
9	346.684708	192.168.1.111	255.255.255.255	DHCP	342	DHCP Inform - Tr
10	346.688406	192.168.1.1	192.168.1.111	DHCP	590	DHCP ACK - Tr
11	430.352810	0.0.0.0	255.255.255.255	DHCP	356	DHCP Request - Tr

> Frame 1: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface 0
 > Ethernet II, Src: Apple_dc:8d:76 (28:cf:da:dc:8d:76), Dst: 01:00:00:00:00:00
 > Internet Protocol Version 4, Src: 192.168.1.111, Dst: 255.255.255.255
 > User Datagram Protocol, Src Port: 68, Dst Port: 68
 > Dynamic Host Configuration Protocol (Inform)

0000 ff ff ff ff ff ff 28 cf da dc 8d 76 08 00
 0010 01 48 08 c0 00 00 11 6e ce c0 a8 01 6f
 0020 ff ff 00 44 00 43 01 34 b1 4c 01 01 06 00
 0030 f9 28 00 00 00 00 c0 a8 01 6f 00 00 00 00
 0040 00 00 00 00 00 00 28 cf da dc 8d 76 00 00
 0050 00 00 00 00 00 00 00 00 00 00 00 00 00
 0060 00 00 00 00 00 00 00 00 00 00 00 00 00
 0070 00 00 00 00 00 00 00 00 00 00 00 00 00
 0080 00 00 00 00 00 00 00 00 00 00 00 00 00
 0090 00 00 00 00 00 00 00 00 00 00 00 00 00
 00a0 00 00 00 00 00 00 00 00 00 00 00 00 00
 00b0 00 00 00 00 00 00 00 00 00 00 00 00 00
 00c0 00 00 00 00 00 00 00 00 00 00 00 00 00
 00d0 00 00 00 00 00 00 00 00 00 00 00 00 00
 00e0 00 00 00 00 00 00 00 00 00 00 00 00 00

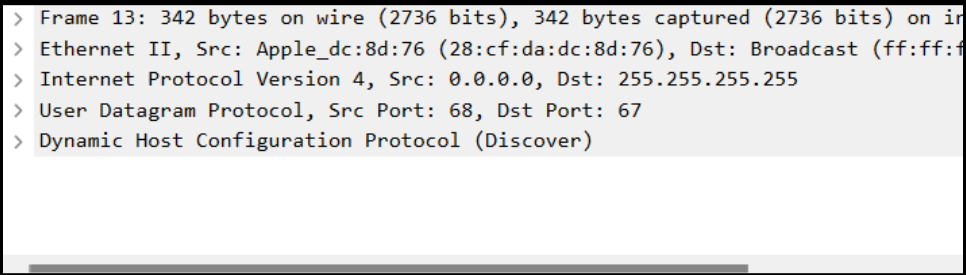
Dynamic Host Configuration Protocol: Protocol | Packets: 41 · Displayed: 41 (100.0%) | Profile: Default

Magic cookie: DHCP

- > Option: (53) DHCP Message Type (Discover)
- > Option: (61) Client identifier
- > Option: (12) Host Name
- > Option: (60) Vendor class identifier
- > Option: (55) Parameter Request List
- > Option: (255) End
- Padding: 000000000000000000

b. Is this DHCP Discover message sent out using UDP or TCP as the underlying transport protocol?

Screenshot:



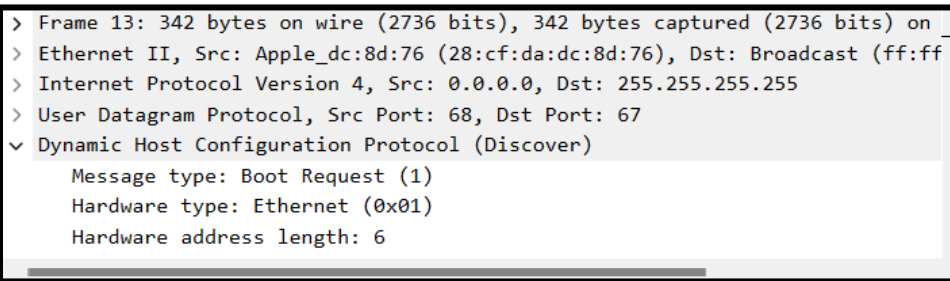
```
> Frame 13: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface  
> Ethernet II, Src: Apple_dc:8d:76 (28:cf:da:dc:8d:76), Dst: Broadcast (ff:ff:ff:ff:ff:ff)  
> Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255  
> User Datagram Protocol, Src Port: 68, Dst Port: 67  
> Dynamic Host Configuration Protocol (Discover)
```

c. What are the source IP address and destination IP address used in the IP datagram containing the Discover message? Is there anything special about this address? Explain.

Theory:

The source IP address used in the IP datagram containing the Discover message is 0.0.0.0 because the host doesn't have an IP address assigned to it yet. The DHCP client broadcasts a DHCPDISCOVER message on the network subnet using the destination address 255.255.255.255.

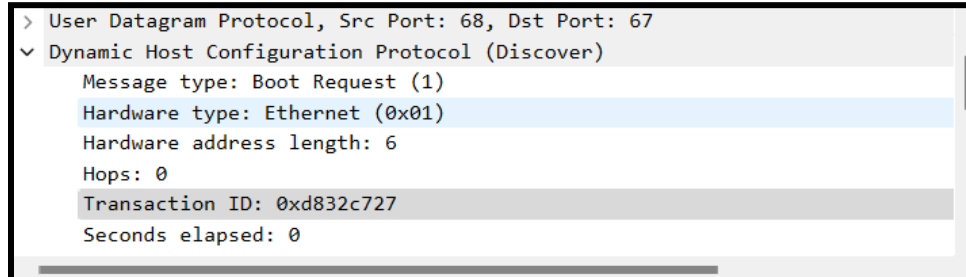
Screenshot:



```
> Frame 13: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface  
> Ethernet II, Src: Apple_dc:8d:76 (28:cf:da:dc:8d:76), Dst: Broadcast (ff:ff:ff:ff:ff:ff)  
> Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255  
> User Datagram Protocol, Src Port: 68, Dst Port: 67  
✓ Dynamic Host Configuration Protocol (Discover)  
  Message type: Boot Request (1)  
  Hardware type: Ethernet (0x01)  
  Hardware address length: 6
```

d. What is the value in the transaction ID field of this DHCP Discover message?

Screenshot:



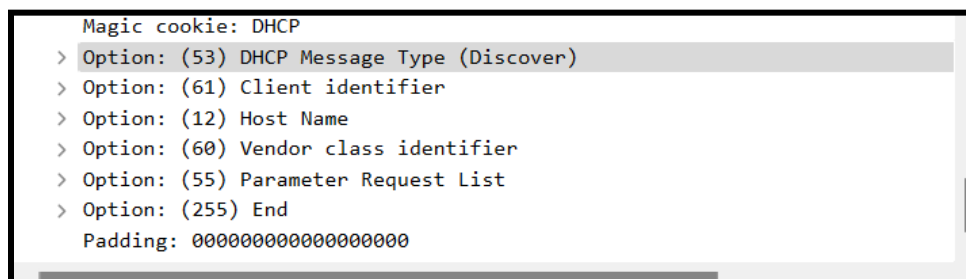
```
> User Datagram Protocol, Src Port: 68, Dst Port: 67
  > Dynamic Host Configuration Protocol (Discover)
    Message type: Boot Request (1)
    Hardware type: Ethernet (0x01)
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0xd832c727
    Seconds elapsed: 0
```

e. Now inspect the options field in the DHCP Discover message. What are five pieces of information (beyond an IP address) that the client is suggesting or requesting to receive from the DHCP server as part of this DHCP transaction?

Theory:

The DHCP options field that the client is asking the DHCP server to provide as part of the DHCP transaction will be: Parameter Request List, Maximum DHCP Message Size, Client Identifier, IP Address Lease Time, Host Name.

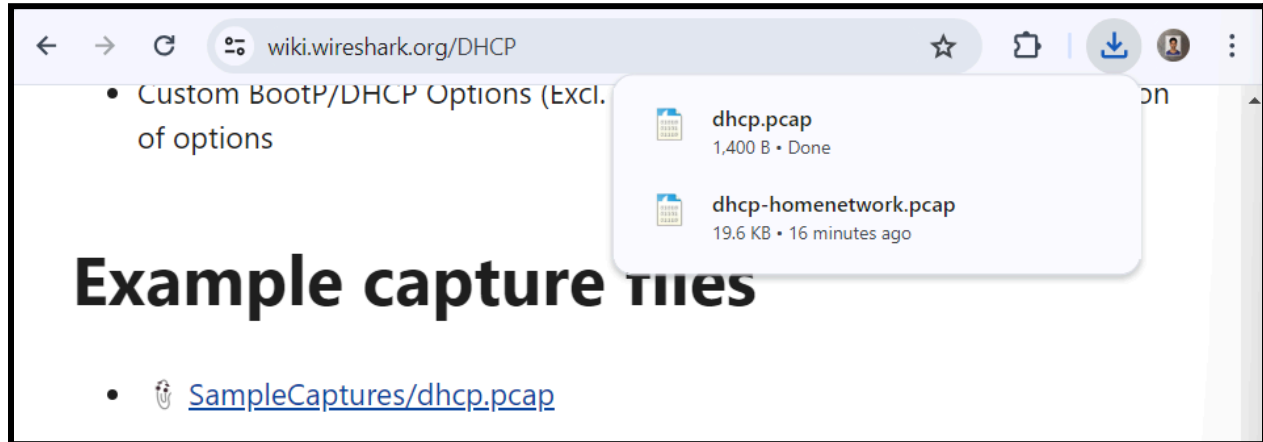
Screenshot:



```
  Magic cookie: DHCP
  > Option: (53) DHCP Message Type (Discover)
  > Option: (61) Client identifier
  > Option: (12) Host Name
  > Option: (60) Vendor class identifier
  > Option: (55) Parameter Request List
  > Option: (255) End
  Padding: 00000000000000000000
```

8. Download the dhcp.pcap file from <https://wiki.wireshark.org/DHCP> and use Wireshark to answer the following questions:

Screenshot:



a. What is the source IP address in the IP datagram containing this ACK message? Is there anything special about this address? Explain.

Theory:

The source IP address in the IP datagram containing this ACK message is 192.168.0.1. This address belongs to the private IP address range reserved for internal networks, as defined in RFC 1918. It's commonly used in home or office networks for devices to communicate internally while being hidden behind a router or firewall. These addresses are not routable over the public internet, which helps improve network security by keeping internal network devices hidden from external threats.

Screenshot:

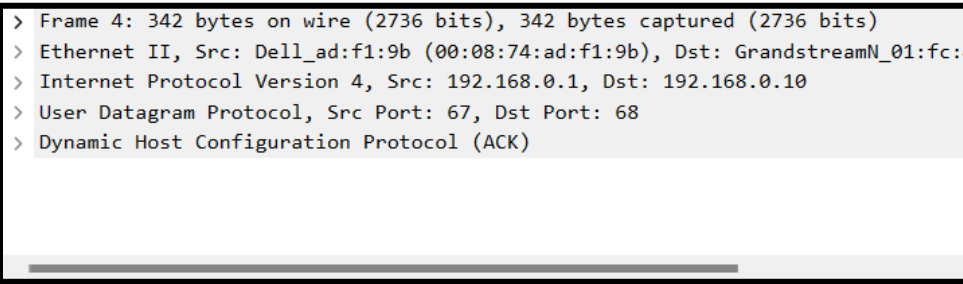
```
> Frame 4: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits)
> Ethernet II, Src: Dell_ad:f1:9b (00:08:74:ad:f1:9b), Dst: GrandstreamN_01:fc:
> Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.10
> User Datagram Protocol, Src Port: 67, Dst Port: 68
> Dynamic Host Configuration Protocol (ACK)
```

b. What is the destination IP address used in the datagram containing this ACK message? Is there anything special about this address? Explain.

Theory:

The destination IP address in the IP datagram containing this ACK message is 192.168.0.10. This address belongs to the same local network (192.168.0.0/24) as the source address (192.168.0.1), indicating that the communication is happening within the same local network. This is often referred to as local or internal communication. It typically doesn't traverse through routers or the internet, making it faster and more secure compared to communication across different networks.

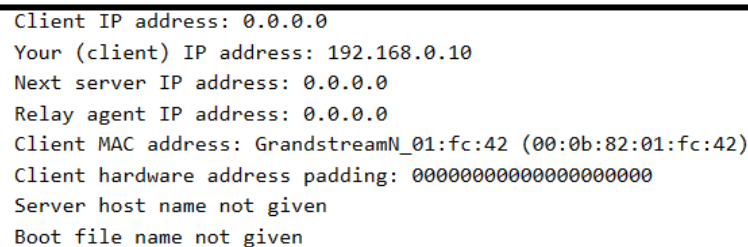
Screenshot:



```
> Frame 4: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface 0
> Ethernet II, Src: Dell_ad:f1:9b (00:08:74:ad:f1:9b), Dst: GrandstreamN_01:fc:42 (00:0b:82:01:fc:42)
> Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.10
> User Datagram Protocol, Src Port: 67, Dst Port: 68
> Dynamic Host Configuration Protocol (ACK)
```

c. What is the name of the field in the DHCP ACK message (as indicated in the Wireshark window) that contains the assigned client IP address?

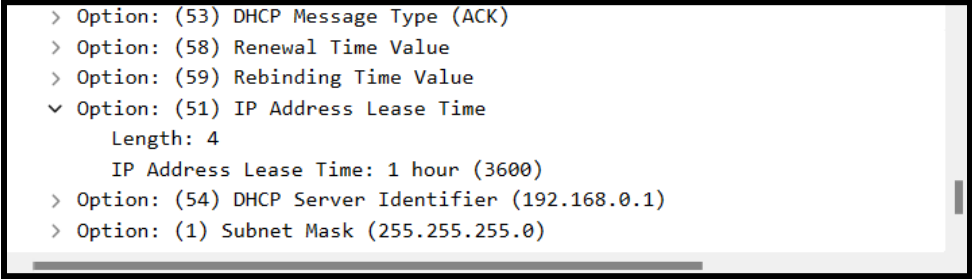
Screenshot:



```
Client IP address: 0.0.0.0
Your (client) IP address: 192.168.0.10
Next server IP address: 0.0.0.0
Relay agent IP address: 0.0.0.0
Client MAC address: GrandstreamN_01:fc:42 (00:0b:82:01:fc:42)
Client hardware address padding: 00000000000000000000
Server host name not given
Boot file name not given
```


d. For how long a time (the so-called "lease time") has the DHCP server assigned this IP address to the client?

Screenshot:

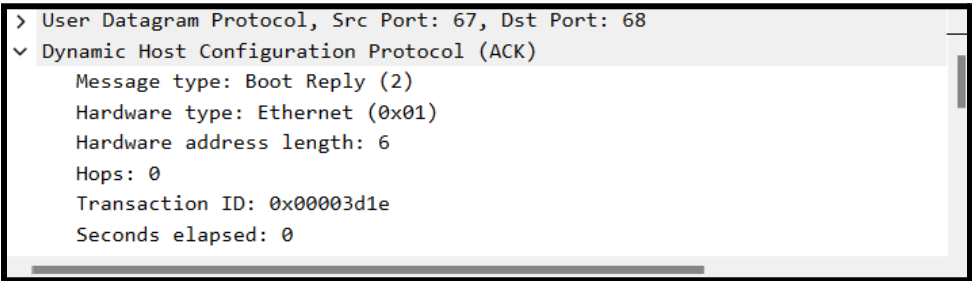


A screenshot of a network tool's DHCP options list. The list is expanded to show 'Option: (51) IP Address Lease Time'. Below this option, it shows 'Length: 4' and 'IP Address Lease Time: 1 hour (3600)'. Other options visible include (53) DHCP Message Type (ACK), (58) Renewal Time Value, (59) Rebinding Time Value, (54) DHCP Server Identifier (192.168.0.1), and (1) Subnet Mask (255.255.255.0).

```
> Option: (53) DHCP Message Type (ACK)
> Option: (58) Renewal Time Value
> Option: (59) Rebinding Time Value
v Option: (51) IP Address Lease Time
    Length: 4
    IP Address Lease Time: 1 hour (3600)
> Option: (54) DHCP Server Identifier (192.168.0.1)
> Option: (1) Subnet Mask (255.255.255.0)
```

e. What is the IP address (returned by the DHCP server to the DHCP client in this DHCP ACK message) of the first hop router on the default path from the client to the rest of the Internet?

Screenshot:



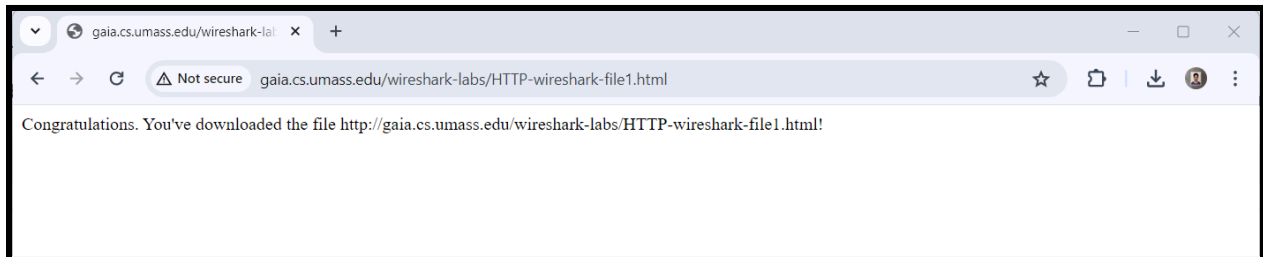
A screenshot of a network tool's DHCP ACK message details. The message is identified as 'Dynamic Host Configuration Protocol (ACK)'. It shows 'Message type: Boot Reply (2)', 'Hardware type: Ethernet (0x01)', 'Hardware address length: 6', 'Hops: 0', 'Transaction ID: 0x00003d1e', and 'Seconds elapsed: 0'.

```
> User Datagram Protocol, Src Port: 67, Dst Port: 68
v Dynamic Host Configuration Protocol (ACK)
    Message type: Boot Reply (2)
    Hardware type: Ethernet (0x01)
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0x00003d1e
    Seconds elapsed: 0
```

9. Use Wireshark to capture packets by browsing any website and answer the following questions based on the contents of the Ethernet frame containing the HTTP GET message from

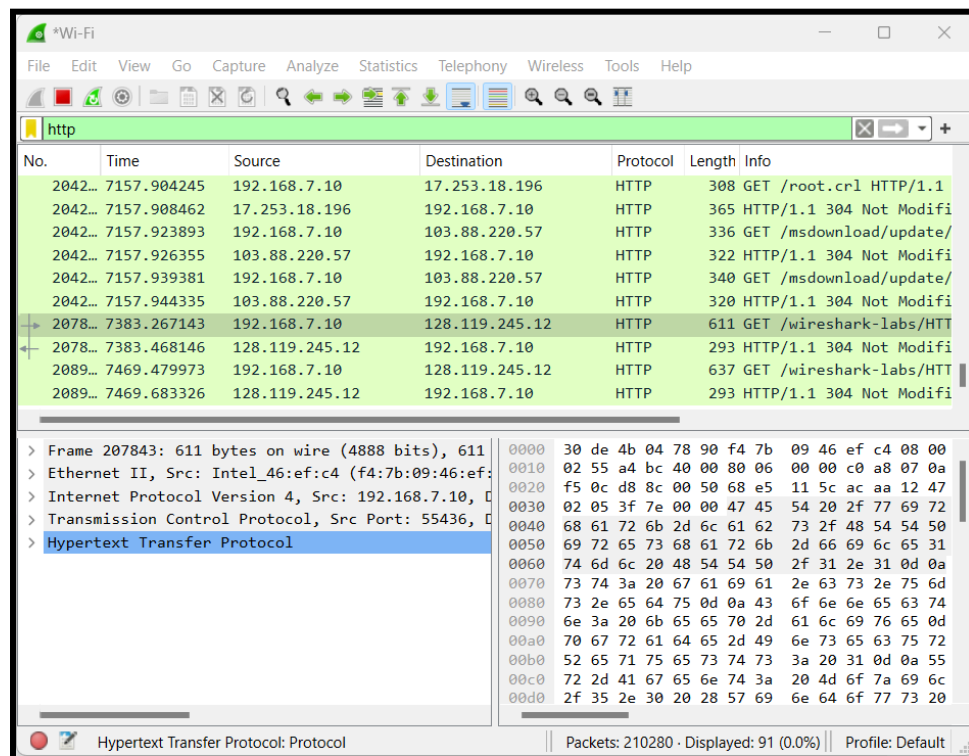
<http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html>:

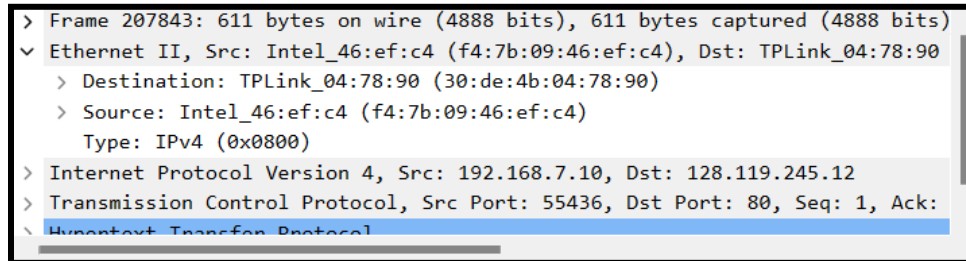
Screenshot:



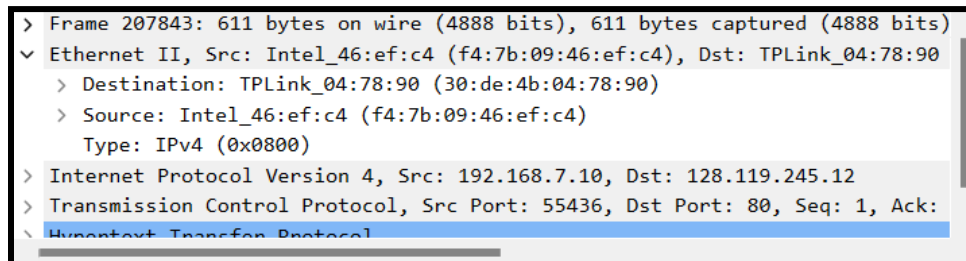
a. Filter the packets using "http." Answer the following questions, based on the contents of the Ethernet frame containing the HTTP GET message.

Screenshot:



b. What is the 48-bit Ethernet address of your computer?**Screenshot:**A screenshot of a Wireshark packet capture window. The packet list on the left shows 'Frame 207843: 611 bytes on wire (4888 bits), 611 bytes captured (4888 bits)'. The packet details pane on the right shows the following layers: Ethernet II, Src: Intel_46:ef:c4 (f4:7b:09:46:ef:c4), Dst: TPLink_04:78:90 (30:de:4b:04:78:90); Destination: TPLink_04:78:90 (30:de:4b:04:78:90); Source: Intel_46:ef:c4 (f4:7b:09:46:ef:c4); Type: IPv4 (0x0800); Internet Protocol Version 4, Src: 192.168.7.10, Dst: 128.119.245.12; Transmission Control Protocol, Src Port: 55436, Dst Port: 80, Seq: 1, Ack: 1; and Hypertext Transfer Protocol. The 'Hypertext Transfer Protocol' layer is highlighted in blue.

```
> Frame 207843: 611 bytes on wire (4888 bits), 611 bytes captured (4888 bits)
v Ethernet II, Src: Intel_46:ef:c4 (f4:7b:09:46:ef:c4), Dst: TPLink_04:78:90
  > Destination: TPLink_04:78:90 (30:de:4b:04:78:90)
  > Source: Intel_46:ef:c4 (f4:7b:09:46:ef:c4)
  Type: IPv4 (0x0800)
> Internet Protocol Version 4, Src: 192.168.7.10, Dst: 128.119.245.12
> Transmission Control Protocol, Src Port: 55436, Dst Port: 80, Seq: 1, Ack:
v Hypertext Transfer Protocol
```

c. What is the 48-bit destination address in the Ethernet frame?**Screenshot:**A screenshot of a Wireshark packet capture window, identical to the one above. It shows the details of frame 207843, highlighting the Ethernet II layer with source and destination MAC addresses, and subsequent layers including IP and TCP.

```
> Frame 207843: 611 bytes on wire (4888 bits), 611 bytes captured (4888 bits)
v Ethernet II, Src: Intel_46:ef:c4 (f4:7b:09:46:ef:c4), Dst: TPLink_04:78:90
  > Destination: TPLink_04:78:90 (30:de:4b:04:78:90)
  > Source: Intel_46:ef:c4 (f4:7b:09:46:ef:c4)
  Type: IPv4 (0x0800)
> Internet Protocol Version 4, Src: 192.168.7.10, Dst: 128.119.245.12
> Transmission Control Protocol, Src Port: 55436, Dst Port: 80, Seq: 1, Ack:
v Hypertext Transfer Protocol
```

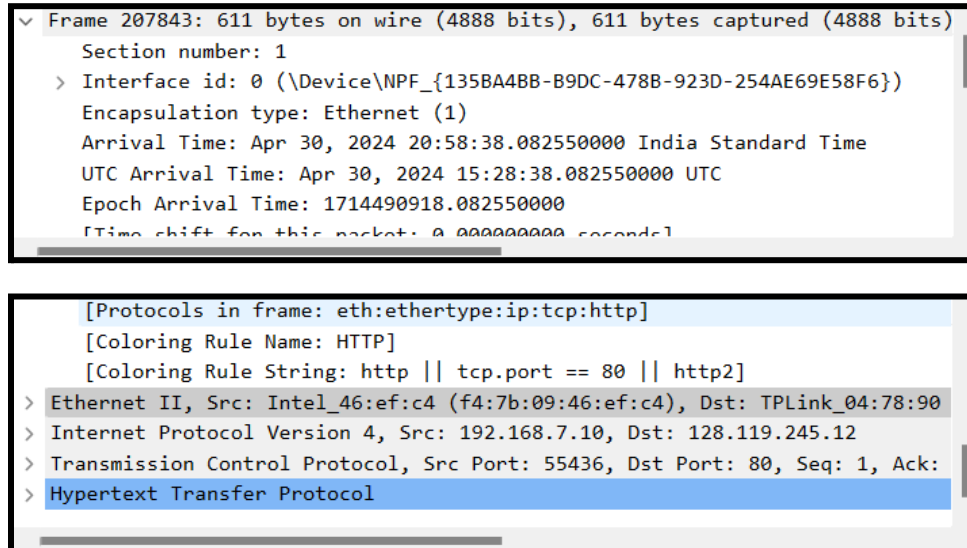
d. Is this the Ethernet address of gaia.cs.umass.edu? What device has this as its Ethernet address?**Theory:**

No, the Ethernet addresses provided are not directly linked to a hostname like "gaia.cs.umass.edu."

The Ethernet address "Intel_46:ef:c4 (f4:7b:09:46:ef:c4)" belongs to a device manufactured by Intel, and the Ethernet address "TPLink_04:78:90 (30:de:4b:04:78:90)" belongs to a device manufactured by TP-Link. Without additional context, it's not possible to determine which specific devices these addresses correspond to, as they could be network interface cards, routers, switches, or any other network-enabled devices.

e. Provide the hexadecimal value for the two-byte Frame type field. What upper layer protocol does this correspond to?

Screenshot:



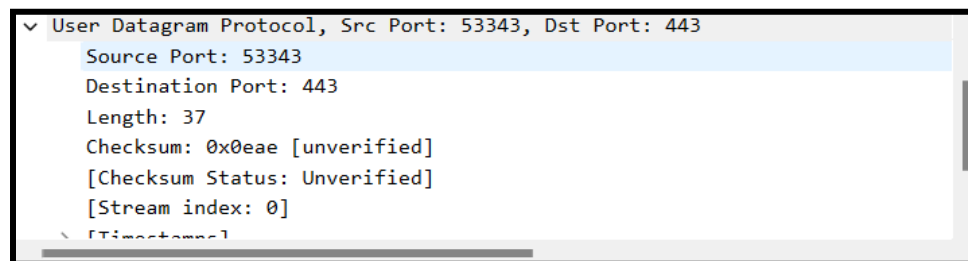
10. Use Wireshark to capture packets by browsing any website and answer the following questions:

a. Select any UDP packet. Find out how many fields are there in the UDP header and name these fields.

Theory:

The UDP header contains 4 fields: source port, destination port, length, and checksum.

Screenshot:



b. Determine the length of each of the UDP header fields.

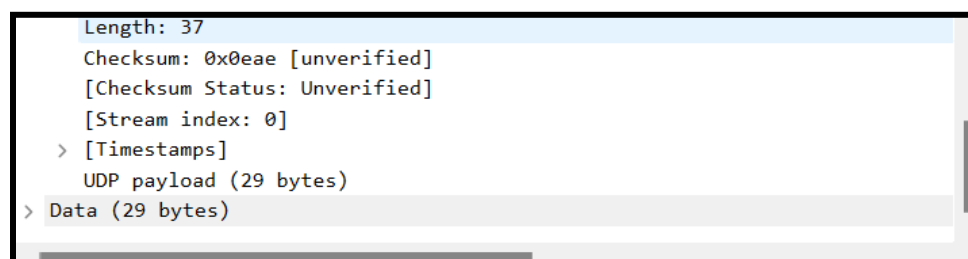
Theory:

From the packet content field, determine the length (in bytes) of each of the UDP header fields.

Each of the UDP header fields is 2 bytes long.

c. What is the value in the length field indicated? What is the length of the UDP payload?

Screenshot:



d. What is the maximum number of bytes that can be included in a UDP payload?

Theory:

The maximum number of bytes that can be included in a UDP payload is $(2^{16} - 1)$ bytes plus the header bytes. This gives $65535 \text{ bytes} - 29 \text{ bytes} = 65506 \text{ bytes}$.

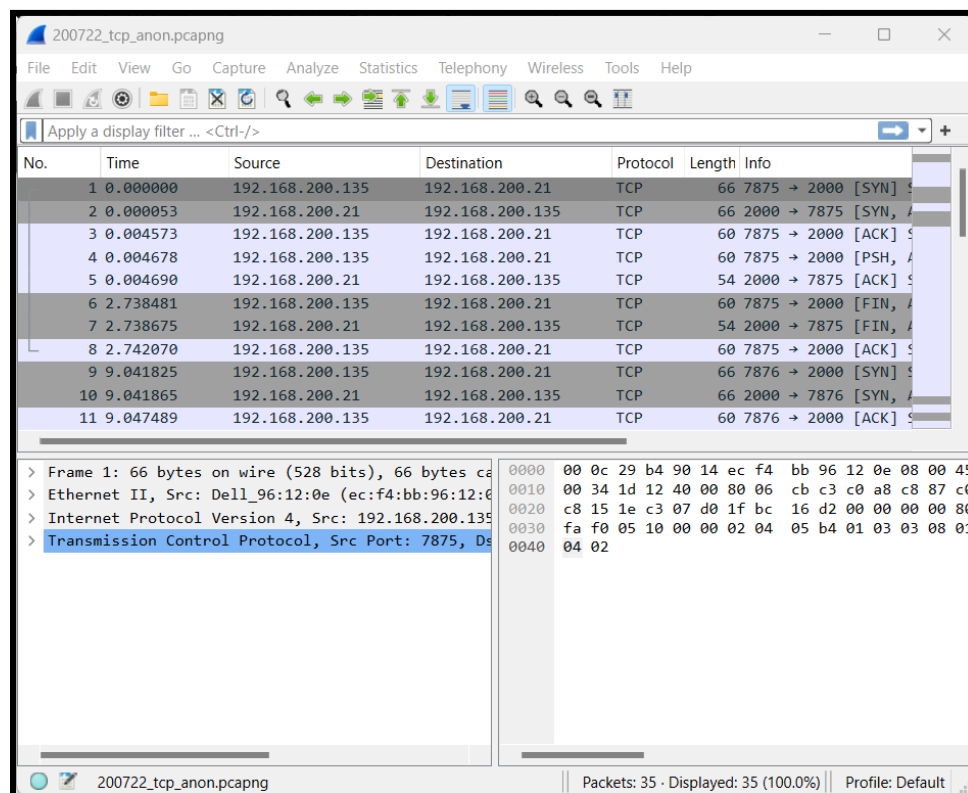
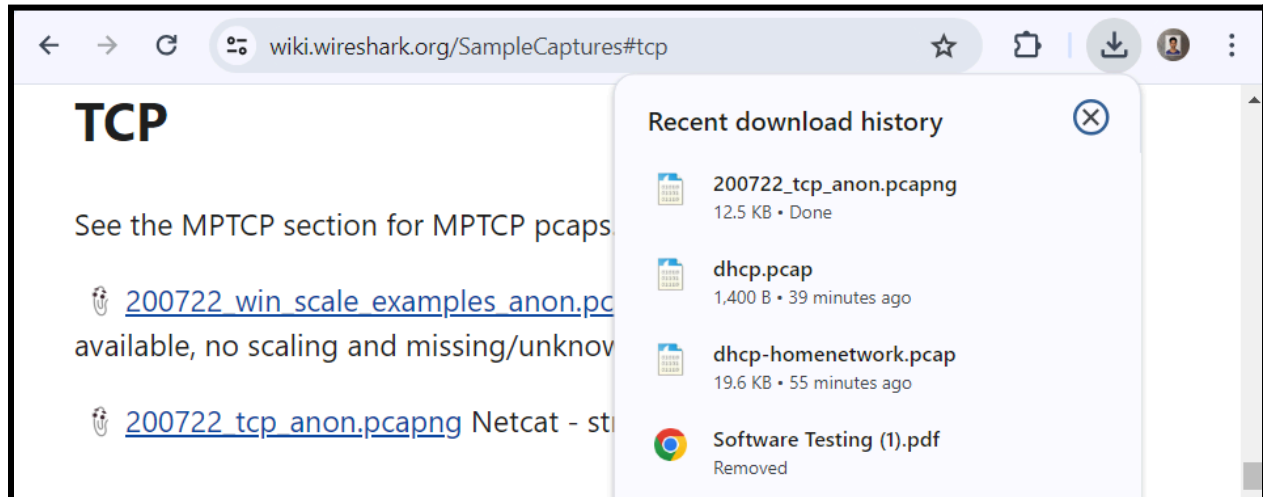
e. What are the largest possible source and destination port numbers?

Theory:

The largest possible source and destination port numbers are both $65535 (2^{16} - 1)$, as the port numbers range from 0 to 65535.

11. From <https://wiki.wireshark.org/SampleCaptures#tcp>, download the 200722_tcp_anon.pcap ng file. Use Wireshark to answer the following questions:

Screenshot:



a. What is the IP address and TCP port number used by the client?**Screenshot:**

```
> Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on inter
> Ethernet II, Src: Dell_96:12:0e (ec:f4:bb:96:12:0e), Dst: VMware_b4:90:14 (
> Internet Protocol Version 4, Src: 192.168.200.135, Dst: 192.168.200.21
> Transmission Control Protocol, Src Port: 7875, Dst Port: 2000, Seq: 0, Len:
  Source Port: 7875
  Destination Port: 2000
  [Stream index: 0]
  > [Conversation completeness: Complete, WITH_DATA (31)]
```

b. What is the IP address and TCP port number used by the server?**Screenshot:**

```
> Frame 2: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on inter
> Ethernet II, Src: VMware_b4:90:14 (00:0c:29:b4:90:14), Dst: Dell_96:12:0e (
> Internet Protocol Version 4, Src: 192.168.200.21, Dst: 192.168.200.135
> Transmission Control Protocol, Src Port: 2000, Dst Port: 7875, Seq: 0, Ack:
  Source Port: 2000
  Destination Port: 7875
  [Stream index: 0]
  > [Conversation completeness: Complete, WITH_DATA (31)]
```

c. What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and the server?**What identifies the segment as a SYN segment?****Screenshot:**

```
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 532420306
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 0
Acknowledgment number (raw): 0
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x002 (SYN)
Window: 64240
```

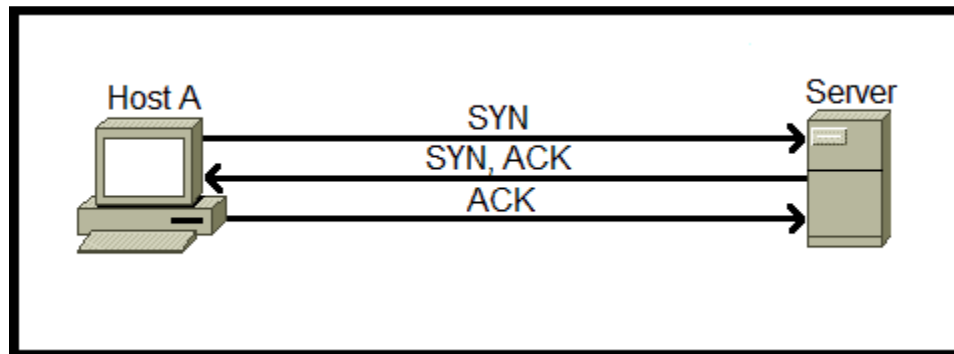

d. What is the sequence number of the TCP SYNACK segment sent by the server to the client computer in reply to the SYN? What is the value of the acknowledgment field in the SYNACK segment?

Screenshot:

```
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 2978637659
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 532420307
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x012 (SYN, ACK)
Window: 64240
```

e. How did the server determine that value? What identifies the segment as a SYNACK segment?

Screenshot:



```
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 532420306
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 0
Acknowledgment number (raw): 0
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x002 (SYN)
Window: 64240
```

```
Sequence Number: 0      (relative sequence number)
Sequence Number (raw): 2978637659
[Next Sequence Number: 1      (relative sequence number)]
Acknowledgment Number: 1      (relative ack number)
Acknowledgment number (raw): 532420307
1000 .... = Header Length: 32 bytes (8)
> Flags: 0x012 (SYN, ACK)
Window: 64240
```

```
Sequence Number: 1      (relative sequence number)
Sequence Number (raw): 532420307
[Next Sequence Number: 1      (relative sequence number)]
Acknowledgment Number: 1      (relative ack number)
Acknowledgment number (raw): 2978637660
0101 .... = Header Length: 20 bytes (5)
> Flags: 0x010 (ACK)
Window: 1026
```