Initial Learning Plan- SIT202 - T1 2024

Study Plan

Tasks	Due Date (P)	Due Date (C)	Due Date (D)	Due Date (HD)
P tasks (Lesson Reviews)				
Task 1.2P Introduction	15 March	15 March	8 March	8 March
Task 3.1P Application Layer	29 March	29 March	22 March	22 March
Task 5.1P Transport Layer	19 April	19 April	12 April	12 April
Task 6.1P Network Layer- Data Plane	26 April	23 April	19 April	19 April
Task 8.1P Network Layer – Control Plane	10 May	10 May	3 May	3 May
Task 9.1P Data Link Layer	17 May	15 May	10 May	10 May
Task 10.1P Physical Layer	24 May	21 May	17 May	17 May
C tasks				
Task 4.1 C Above and Beyond Pass	Not targeted	26 April	19 April	19 April
Task 5.2 C My DNS Server Sketch	Not targeted	12 April	9 April	9 April
Task 6.2 C My DNS Server	Not targeted	26 April	19 April	19 April
D tasks				
Task 5.2 D Above and Beyond Credit	Not targeted	Not targeted	17 May	17 May
Task 7.2D Amazing Networks	Not targeted	Not targeted	3 May	30 April
HD tasks				
Task 1.3 H Collaborative Learning Initiatives & Outcomes (aim for exemplary achievement 90+)	Not targeted	Not targeted	Not targeted	24 May
Task 6.3 H My Awesome Tutorial	Not targeted	Not targeted	Not targeted	3 May
Task 8.2 H Building Your Own Ping Program	Not targeted	Not targeted	Not targeted	17 May

Grading Agreement

I, (Full name, Student ID), confirm that I have read and understood how this unit works: I understand that:

- I will receive a grade in accordance with the number of modules and tasks that I complete;
- I can change my target grade at any time during the trimester, up until I submit my portfolio.
- I am responsible for providing evidence of my work to satisfy the requirements in each module completed.
- Following my initial submission, I will work with my tutor and the teaching team to update and complete my submissions until they demonstrate that I have achieved the learning objectives of the module.

I commit to submit the initial submission of my work before the deadlines listed in the table above. I acknowledge that these deadlines can be modified by me, in agreement with my tutor, when the necessity occurs, and that I am responsible for requesting these modifications as soon as I become aware of the need to do so, and no later than the deadline to be modified.

- Name: J. Kenisha Corera
- Student ID: DFCS|DK|62|203
- Date: 2024.07.17
- Signature: J.K.Corera

Computer Network and communication

1.2P

1. Module Summary

- Main Concept
  An overview of the communication protocols that facilitate the exchange of data between network devices. An in-depth analysis of the TCP/IP model's five layers: Application, Transport, Network, Data-Link, and Physical.

- Practical network Analysis
  Play a role-playing exercise that shows how the various TCP/IP model levels interact with one another during communication. An analysis of a browser's resource requests using YouTube serves as a practical example, demonstrating the use of the HTTP/1.1 and H2 protocols. Utilizing network analysis techniques to monitor the total load time and resource load order.

- Detailed Web page Analysis
  Examination of the httpforever.com website, with an emphasis on a number of topics, including: utilized protocols (TCP, HTTP, DNS, OCSP). The duration of requests and answers. determining the web server's IP address. utilizing the developer tools in Chrome to confirm comparable data.

- Important thing
  The capacity to do useful network analysis and have a thorough comprehension of the TCP/IP paradigm are the two most important learning outcomes. This information is essential for maximizing site performance and guaranteeing effective data transfer.

Reflection Content

- When considering the Introduction Module, the most important lessons learned were the thorough comprehension of network communication protocols, especially the TCP/IP model, and the useful use of network analysis tools. These abilities are highly transferable to real-world situations and are essential for locating and fixing performance bottlenecks as well as guaranteeing effective web performance.
- With the use of real-world, hands-on network analysis expertise, this session expands on past understanding of fundamental networking concepts. An excellent basis for evaluating and improving web performance has been established by the thorough investigation of tools such as the Network panel in Developer Tools. This skill set is extremely valuable in the IT sector.
- The goal of the course team is very clear: to give students real-world skills that they can use right away in the digital sector. Students can enhance web application performance and become significant assets in their future employment by learning and putting these approaches to use in their studies. This is in line with the industry's need for experts who can guarantee effective online performance and data transmission, improving employability and career advancement.

Computer networks and communication

1.2P

Evidence of Learning

These are the evidence that for the team work for activity one

Device 1 (Source) - Laptop

Nishad
Application Layer – Nishad

Hi my name is Nishad and I am the Application Layer
As the application layer, I interact with the user and provide network services to applications.
The protocol I follow is HTTP
I have received the message "Hello" from the user and I'm passing this to the transport layer using HTTP

Transport Layer – Raaid

Hi my name is Raaid and I am the Transport Layer.
As the Transport layer I ensure reliable data transfer, flow control, and error checking.
The two main protocols I follow is TCP and UDP.
I break down data to Segments.
I Received "Hello" from the application layer. Adding TCP headers thus beginning data encapsulation I pass it to the network layer.

Kenisha CICRA
Network Layer – Kenisha

Hi my name is Kenisha and I am the Network Layer.
My role is to determine the best path to route the data.
I add IP headers and determine the destination IP address. My job is to route packets throughout the network.
I have received "hello" with TCP headers. I shall now Add IP headers and rout to the data link layer.

Shekaina CICRA
Data Link Layer - Shekaina
Hi I'm Shekaina and I manage node-to-node data transfer and handle errors.
Responsibilities:
- Add MAC addresses.
- Frame the data.
- Manage access to the physical medium.

Received "Hello" with IP headers. Adding MAC addresses, framing the data, and passing to the physical layer.

Device 2 (Destination) – Server

Pavithran AIR
Physical Layer – Pavithran

Hey my name is Pavithran and I am the physical layer.
I handle the actual transmission of data over the physical medium.
I am responsible in converting framed data into signals.
And Transmitting signals over the network medium.
I have received the framed data from the Data Link layer and I converting them into bits and transmitting to Device 2.

Data Link Layer – Shekaina
Hey my name is Shekaina and I am the Data Link layer.
I received the framed data from the physical layer and I am removing MAC addresses and passing to the network layer

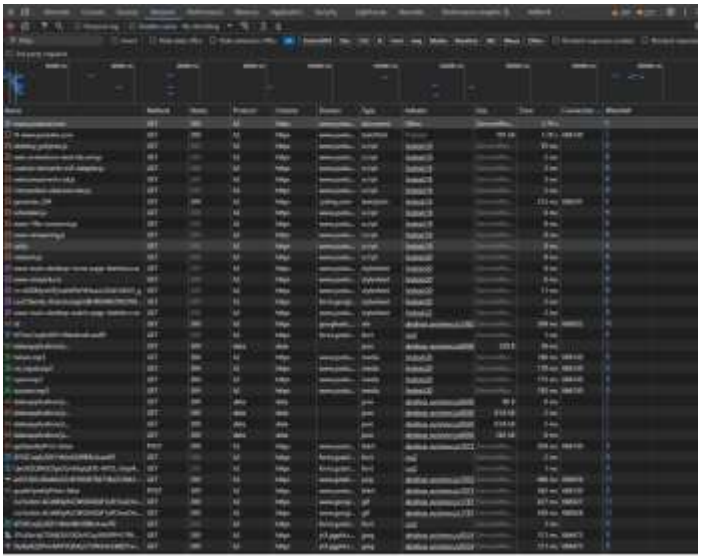
Pavithran AIR
Physical Layer – Pavithran

Hey my name is Pavithran and I am the physical layer.
I Received signals from Device 1 and I am converting back to framed data and passing to the data link layer.

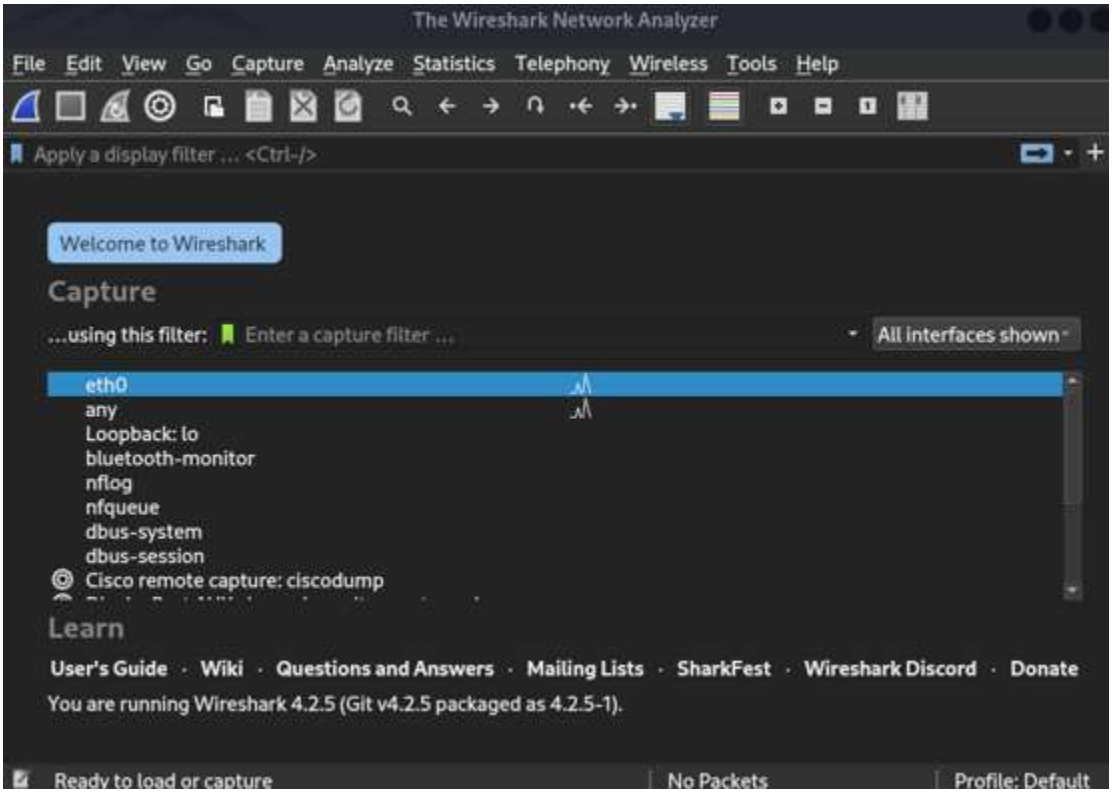
Kenisha CICRA
Network Layer – Kenisha

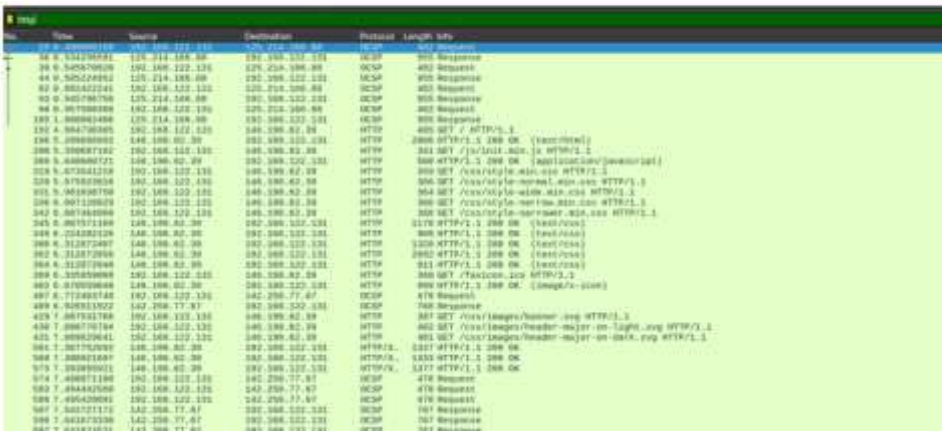
Hi my name is Kenisha and I am the Network layer.
I have Received packets. Removing IP headers and passing segments to the transport layer.

These are the screen shot for Activity task 2



These are the evidence for activity 3





SIT202: Computer Networks and Communication
Leaning Evidence for Active Class Task 3

Name: Kenisha Corera
Student ID: C23020001

Members in this group activity task:

Nishad – C23110001
Kenisha – C23020001
Shekaina – C23110002
Raaid – C23020004
Pavithran – C22060015

Activity 1

1. What is the core Internet function provided by DNS?
    - Domain Name System (DNS) translates human – readable domain names into IP addresses which are used by computers to identify each other in a network.

2. Why do we need DNS?
    - DNS is essential because it simplifies the process of accessing websites and other resources on the internet.

3. What is the layer that DNS belong to?
    - DNS protocol belongs and operates in the Application Layer in the TCP/IP model.

4. Do you think a single DNS server is enough to support the entire network? Justify your answer. Provide alternate solution if we have any
    - A singe DNS server is not sufficient to support the entire network due to the sheer volume of request and the need for redundancy. A distributed system of DNS servers is used to handle the load, provide faster response times and to ensure reliability. Alternatives could include using either Top – Level Domain (TLD) and authoritative name servers.

5. Discuss the steps involved in your browser to send a HTTP request message to the Web server, deakin.edu.au. Assume that this is the first time you access this webpage. You can continue the discussion as a role play.
    a. One group member can act as the web server, another as DNS servers (you need to decide how many DNS serves will be involved), and the remaining members can be the clients.

Client 1 - Nishad
DNS Server 1 - Kenisha
DNS Server 2 - Shekaina
Authoritative DNS Server - Raaid
Web Server (deakin.edu.au) - Pavithran

b. First, Client 1 needs to view deakin.edu.au and initiate the conversation saying the right message to the right device (to the person who is acting as the right device). Use your knowledge about web browsing that we covered in Module 1 and the first part of Module 2. Assume that there is no DNS caching available.
c. All the devices need to respond to each other with the correct messages in the right sequence. Make sure you record all the steps as you will need those notes to complete the activity 2.

Part 1: Client 1 Accessing deakin.edu.au

Shekaina CICRA
DNS Server 2: Sends the IP address back to client 1.

Kenisha CICRA
DNS Server 1: Sends the IP address back to Client 1.

Nishad
Client 1: Great, I have the IP address. Now I can send an HTTP request. 5:00 PM

Client 1: Sends an HTTP request to Web Server at IP address 128.184.204.21: GET / HTTP/1.1 Host: deakin.edu.au 5:00 PM

Pavithran
Web Server: Receives the request, processes it, and sends back an HTTP response. 5:01 PM

Web Server: Sends an HTTP response to Client 1: "HTTP/1.1 200 OK Content-Type: text/html ... [HTML content]" 5:02 PM

d. Now, after Client 1 accessed deakin.edu.au, Client 2 also needs to view deakin.edu.au. Discuss the steps involved in Client 2's web browser to be able to send a HTTP request message to the Web server.

Client 2 - Raaid
DNS Server 1 - Kenisha
DNS Server 2 - Shekaina
Authoritative DNS Server - Nishad
Web Server (deakin.edu.au) - Pavithran

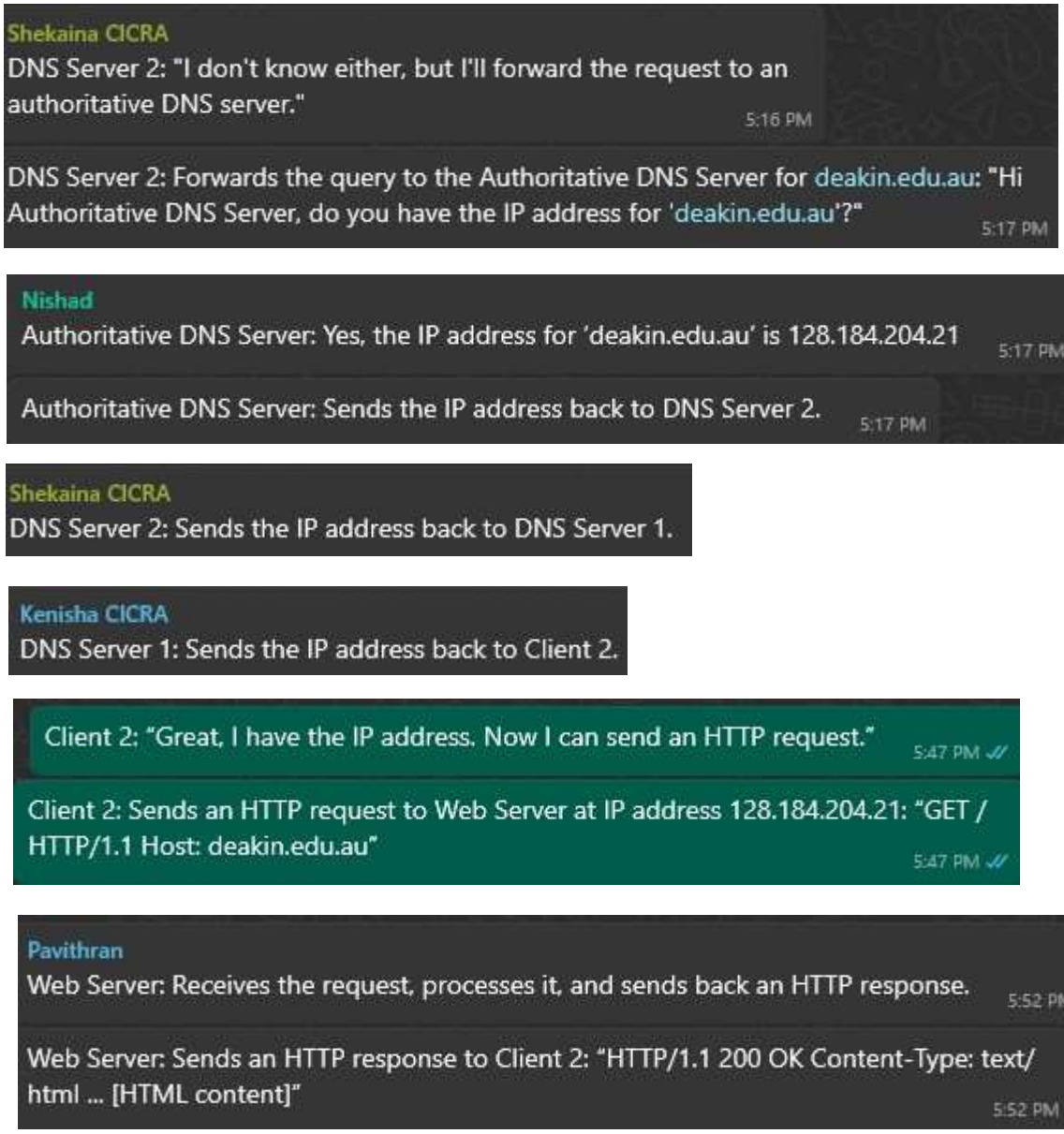
Part 2: Client 2 Accessing 'deakin.edu.au'

Client 2: "I want to visit 'deakin.edu.au'. I need to find its IP address. I'll start by querying the DNS resolver on my system." 5:03 PM

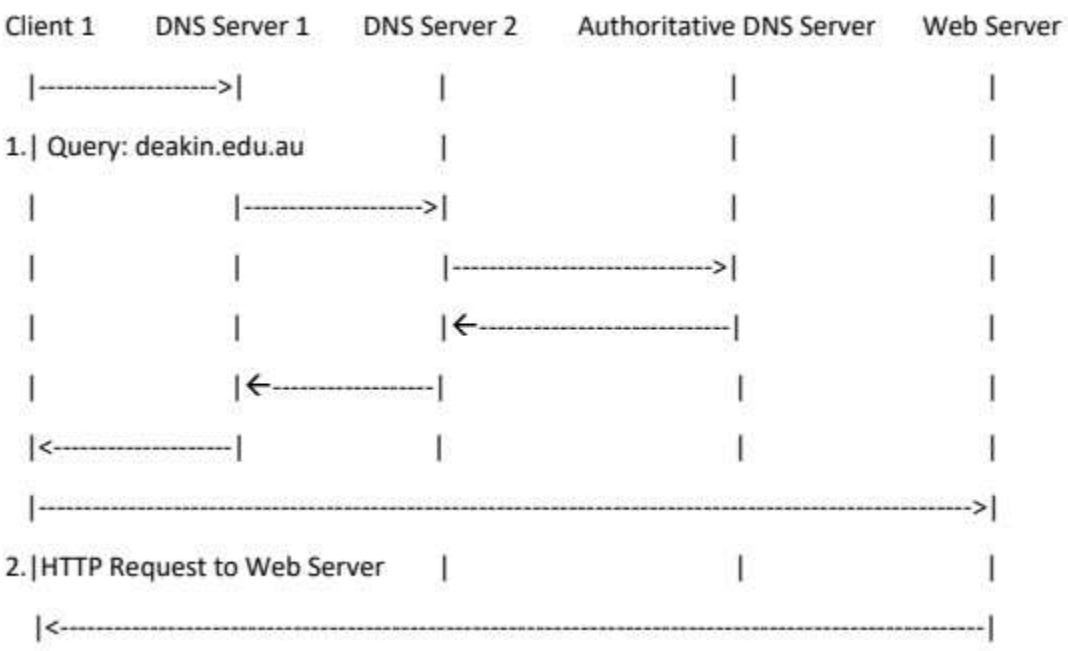
Client 2: Sends a DNS query to DNS Server 1: "Hi DNS Server 1, what's the IP address for 'deakin.edu.au'?" 5:03 PM

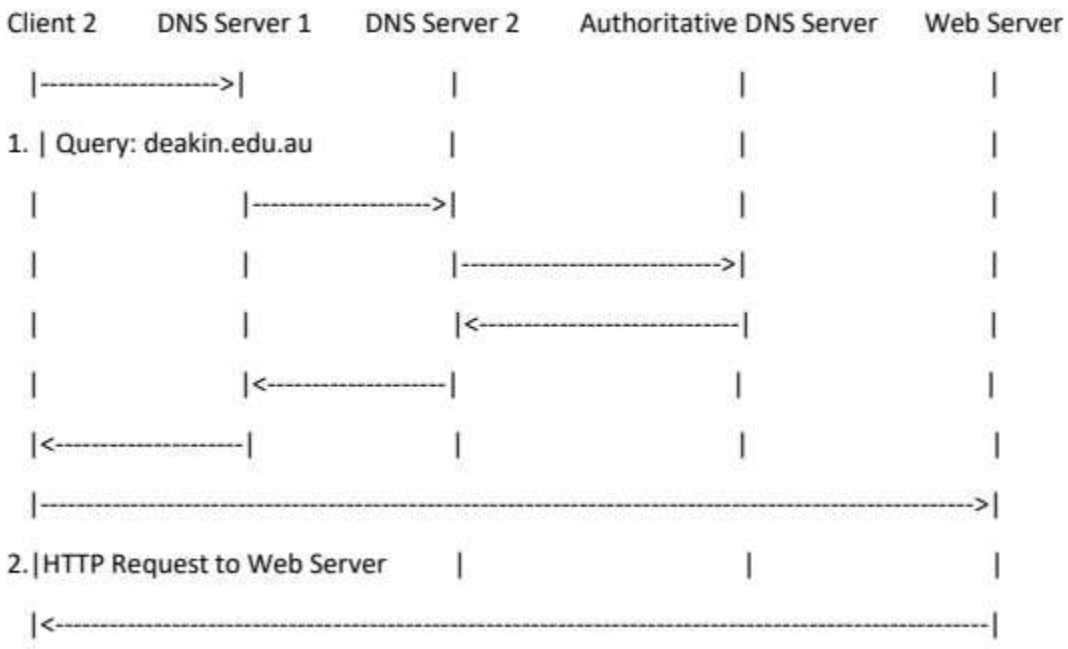
Kenisha CICRA
DNS Server 1: "I don't have that information, but I'll forward your request to another DNS server." 5:04 PM

DNS Server 1: Forwards the query to DNS Server 2: "Hi DNS Server 2, do you know the IP address for 'deakin.edu.au'?" 5:04 PM



e. You can show all the steps in a timing diagram with the end systems and numbering the sequence of steps (similar to the activity you did in Active Class 1).





Activity 2

1. You can send a DNS query message directly to some DNS servers. For this, we use "nslookup"

2. First use “nslookup” in command prompt in Windows or terminal in MacOS to identify the IP address of a Web server deakin.edu.au (in Australia). Note down the webserver and the IP address of that server.

- Webserver: pfSense.home.arpa
  IP Address: 172.16.10.250 and 128.184.20.21

3. What are the answers you received from nslookup?

```
C:\Windows\System32>nslookup deakin.edu.au
Server:  pfSense.home.arpa
Address:  172.16.10.250

Non-authoritative answer:
Name:    deakin.edu.au
Addresses:  2402:6940:1201:2023:128:184:235:77
          2402:6940:1201:2022:128:184:233:77
          2402:6940:1401:2025:128:184:239:77
          2402:6940:1401:2024:128:184:237:77
          128.184.204.21
          128.184.20.21
```

4. What are the authoritative and non-authoritative answers?
   - An authoritative answer comes from a nameserver that is considered authoritative(main) for the domain where as a non-authoritative answer is when the answer is not fetched from the authoritative DNS sever.

5. Use nslookup to identify the authoritative DNS servers for a webserver of a university in USA.

```
Server:  pfSense.home.arpa
Address:  172.16.10.250

Non-authoritative answer:
harvard.edu     nameserver = a1-171.akam.net
harvard.edu     nameserver = a16-64.akam.net
harvard.edu     nameserver = a6-66.akam.net
harvard.edu     nameserver = a7-65.akam.net
harvard.edu     nameserver = a11-67.akam.net
harvard.edu     nameserver = a10-66.akam.net
```

6. Compare the answers in 2 and 5.
   - The nslookup command provides the authoritave DNS servers for Harvard University’s domain whilst the nslookup command in question 2 provides both IPv6 and IPv4 addresses for Deakin University’s domain. So, whilst Harvard university provides the DNS servers that are authoritative for the ‘harvard.edu’ domain, Deakin Universities output displays the list of IP addresses that can be used to access the ‘deakin.edu.au’ web server.

7. Let’s trace DNS in Wireshark now. First,

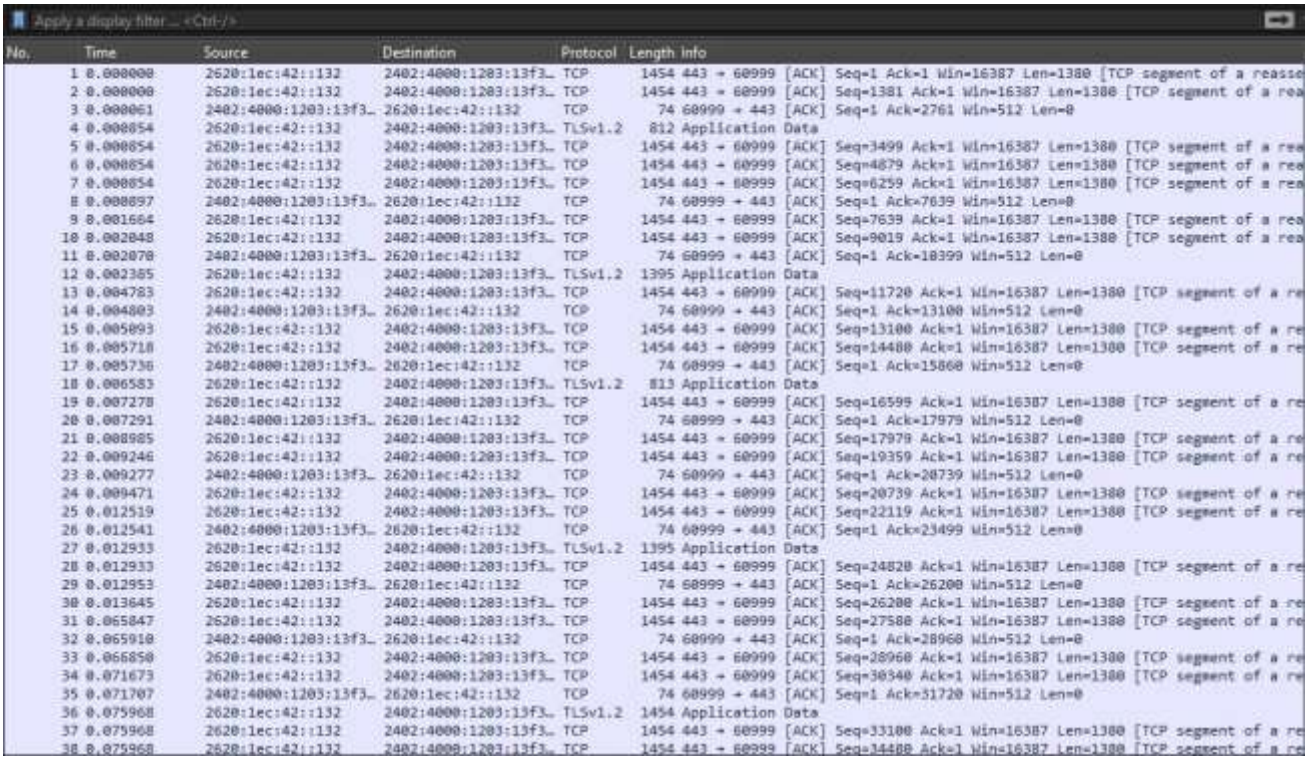
a. Use ipconfig in your command prompt/ terminal to empty the DNS cache in your host ipconfig /flushdns. (MacOS Mojave use: sudo killall -HUP mDNSResponder)

```
C:\Windows\System32>ipconfig /flushdns

Windows IP Configuration

Successfully flushed the DNS Resolver Cache.
```

b. Open google chrome and empty your browser cache.

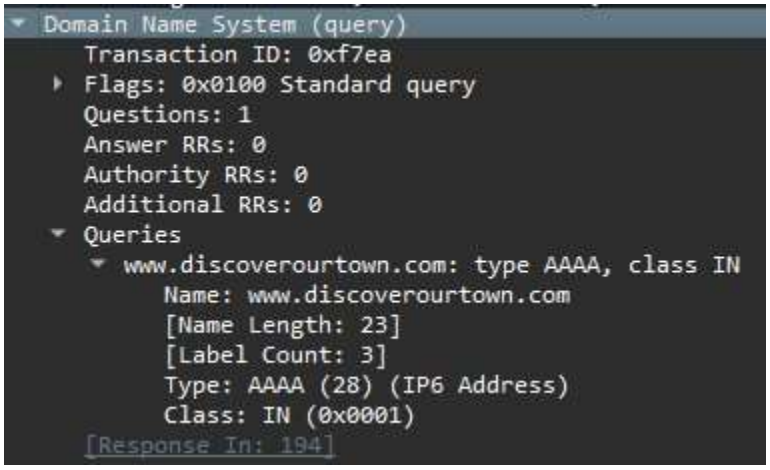
Browser cache emptied.

c. Open Wireshark and start packet capture. Then, visit the Web page: http://www.discoverourtown.com in your browser and stop packet capture.

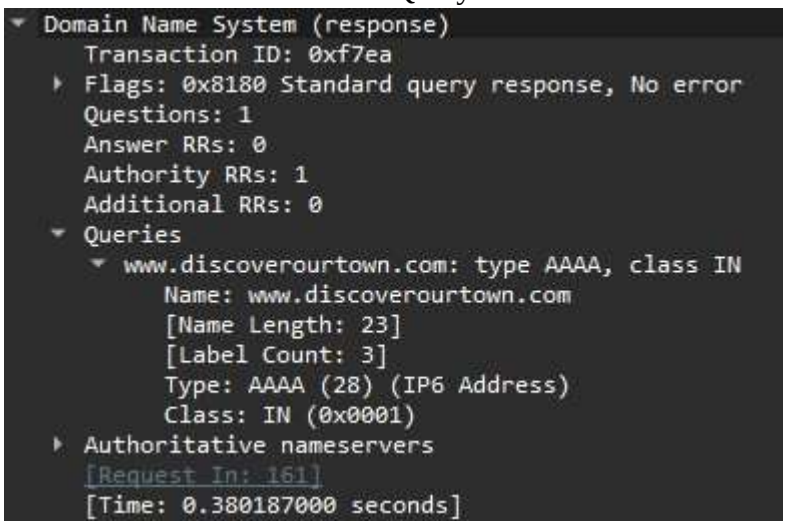


No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=1 Ack=1 Win=16387 Len=1380 [TCP segment of a reasse
2	0.000000	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=1381 Ack=1 Win=16387 Len=1380 [TCP segment of a rea
3	0.000061	2402:4000:1203:13f3…	2620:1ec:42::132	TCP	74	60999 → 443 [ACK] Seq=1 Ack=2761 Win=512 Len=0
4	0.000854	2620:1ec:42::132	2402:4000:1203:13f3…	TLSv1.2	812	Application Data
5	0.000854	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=3499 Ack=1 Win=16387 Len=1380 [TCP segment of a rea
6	0.000854	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=4879 Ack=1 Win=16387 Len=1380 [TCP segment of a rea
7	0.000854	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=6259 Ack=1 Win=16387 Len=1380 [TCP segment of a rea
8	0.000897	2402:4000:1203:13f3…	2620:1ec:42::132	TCP	74	60999 → 443 [ACK] Seq=1 Ack=7639 Win=512 Len=0
9	0.001664	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=7639 Ack=1 Win=16387 Len=1380 [TCP segment of a rea
10	0.002048	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=9019 Ack=1 Win=16387 Len=1380 [TCP segment of a rea
11	0.002070	2402:4000:1203:13f3…	2620:1ec:42::132	TCP	74	60999 → 443 [ACK] Seq=1 Ack=10399 Win=512 Len=0
12	0.002385	2620:1ec:42::132	2402:4000:1203:13f3…	TLSv1.2	1395	Application Data
13	0.004783	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=11720 Ack=1 Win=16387 Len=1380 [TCP segment of a re
14	0.004803	2402:4000:1203:13f3…	2620:1ec:42::132	TCP	74	60999 → 443 [ACK] Seq=1 Ack=13100 Win=512 Len=0
15	0.005093	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=13100 Ack=1 Win=16387 Len=1380 [TCP segment of a re
16	0.005718	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=14480 Ack=1 Win=16387 Len=1380 [TCP segment of a re
17	0.005736	2402:4000:1203:13f3…	2620:1ec:42::132	TCP	74	60999 → 443 [ACK] Seq=1 Ack=15860 Win=512 Len=0
18	0.006583	2620:1ec:42::132	2402:4000:1203:13f3…	TLSv1.2	813	Application Data
19	0.007278	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=16599 Ack=1 Win=16387 Len=1380 [TCP segment of a re
20	0.007291	2402:4000:1203:13f3…	2620:1ec:42::132	TCP	74	60999 → 443 [ACK] Seq=1 Ack=17979 Win=512 Len=0
21	0.008985	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=17979 Ack=1 Win=16387 Len=1380 [TCP segment of a re
22	0.009246	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=19359 Ack=1 Win=16387 Len=1380 [TCP segment of a re
23	0.009277	2402:4000:1203:13f3…	2620:1ec:42::132	TCP	74	60999 → 443 [ACK] Seq=1 Ack=20739 Win=512 Len=0
24	0.009471	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=20739 Ack=1 Win=16387 Len=1380 [TCP segment of a re
25	0.012519	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=22119 Ack=1 Win=16387 Len=1380 [TCP segment of a re
26	0.012541	2402:4000:1203:13f3…	2620:1ec:42::132	TCP	74	60999 → 443 [ACK] Seq=1 Ack=23499 Win=512 Len=0
27	0.012933	2620:1ec:42::132	2402:4000:1203:13f3…	TLSv1.2	1395	Application Data
28	0.012933	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=24820 Ack=1 Win=16387 Len=1380 [TCP segment of a re
29	0.012953	2402:4000:1203:13f3…	2620:1ec:42::132	TCP	74	60999 → 443 [ACK] Seq=1 Ack=26200 Win=512 Len=0
30	0.013645	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=26200 Ack=1 Win=16387 Len=1380 [TCP segment of a re
31	0.065847	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=27580 Ack=1 Win=16387 Len=1380 [TCP segment of a re
32	0.065910	2402:4000:1203:13f3…	2620:1ec:42::132	TCP	74	60999 → 443 [ACK] Seq=1 Ack=28960 Win=512 Len=0
33	0.066850	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=28960 Ack=1 Win=16387 Len=1380 [TCP segment of a re
34	0.071673	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=30340 Ack=1 Win=16387 Len=1380 [TCP segment of a re
35	0.071707	2402:4000:1203:13f3…	2620:1ec:42::132	TCP	74	60999 → 443 [ACK] Seq=1 Ack=31720 Win=512 Len=0
36	0.075968	2620:1ec:42::132	2402:4000:1203:13f3…	TLSv1.2	1454	Application Data
37	0.075968	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=33100 Ack=1 Win=16387 Len=1380 [TCP segment of a re
38	0.075968	2620:1ec:42::132	2402:4000:1203:13f3…	TCP	1454	443 → 60999 [ACK] Seq=34480 Ack=1 Win=16387 Len=1380 [TCP segment of a re

8. Now you are ready to analyze what you captured in Wireshark and explore more about DNS. Use the following questions as a guide for your analysis.
   a. Find the DNS query and response messages. Which transport layer protocol they have used? UDP or TCP?



DNS Query



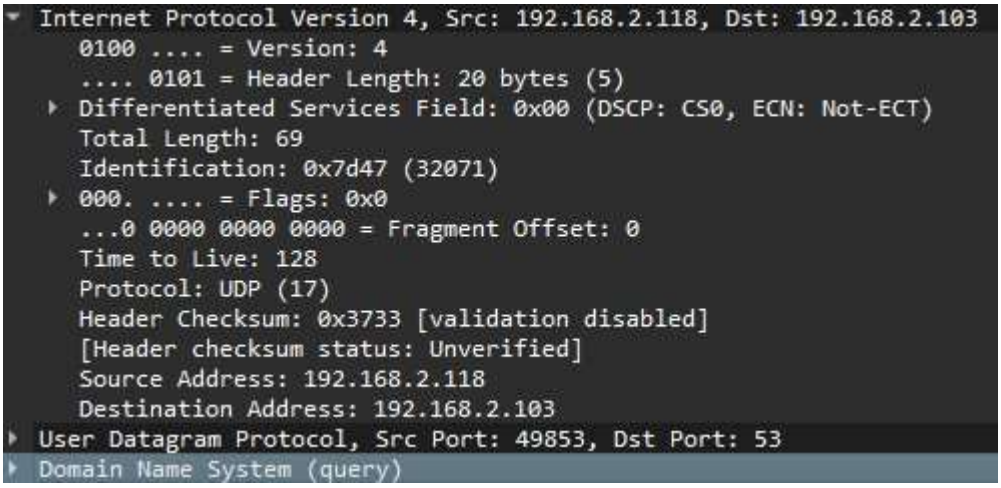
DNS Response

They both use UDP.

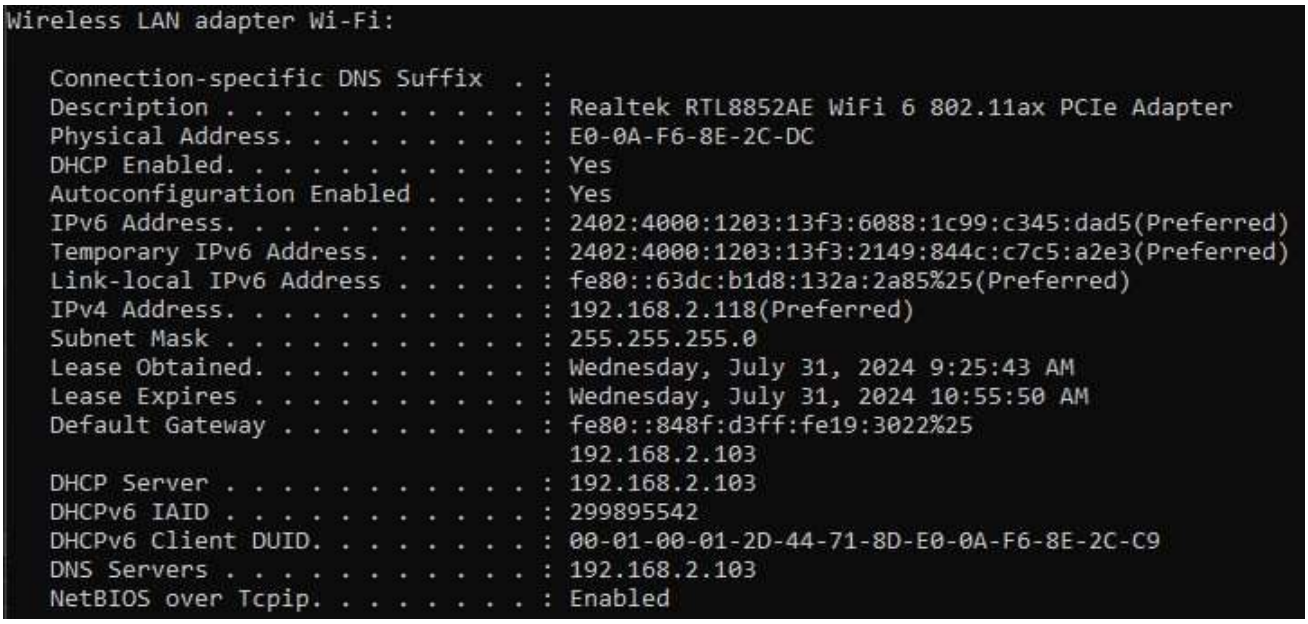
b. What are the destination port of the DNS query message and the source port of DNS response message?

The destination port of query port and and source port of response messages are both 53 because DNS queries are sent through UDP port number 53.

c. What is the IP address that the DNS query message was sent?



d. Identify the IP address of your local DNS server using the terminal/command prompt. Are these two IP addresses identified in c and d the same?

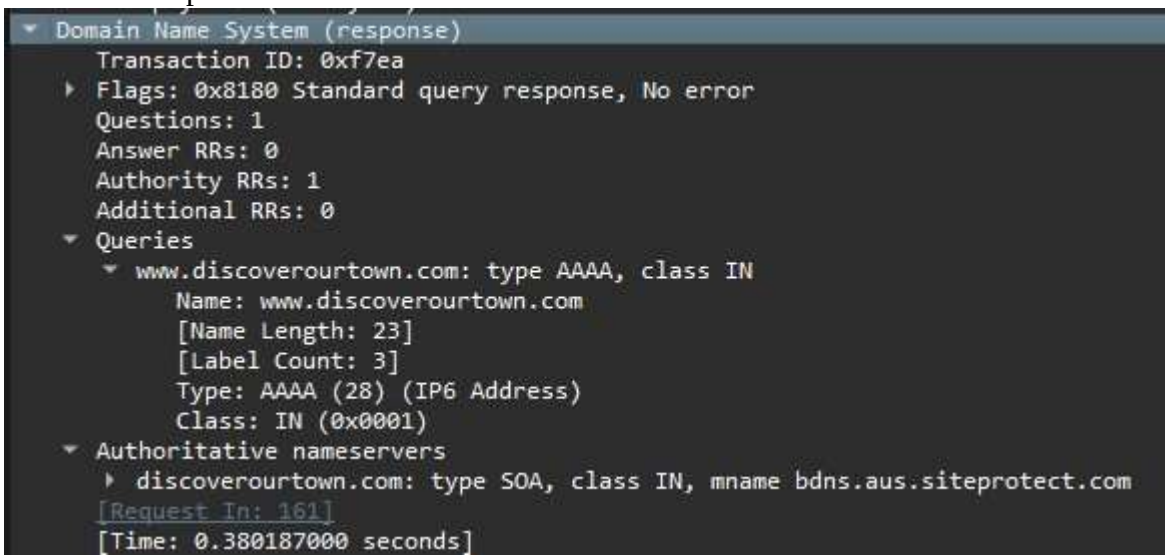


They have the same IP address

e. You can further explore the DNS query message. Can you identify the “Type” of DNS query? What does the query message contain?

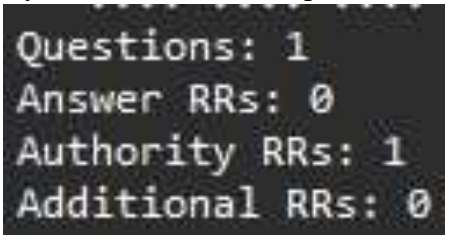
The DNS query message can be seen above and its type is AAAA

f. You can further explore the DNS response message. Can you identify the "Type" of DNS response?



The DNS response message can be seen above and its type is AAAA

g. Are there any "answers" in the response message? If so, what are these answers?



As you can see there are no answers in the response message.

h. The web page that you have accessed contains a couple of images. Does the host request new DNS queries to access each image? Explain your answer.

The host does not need to necessarily request new DNS queries for images that are located in the same domain as is the case for this webpage.

Above and Beyond Tasks

- Explain E-mail (another popular application) o What is the principal application layer protocol used in e-mails?
    - The protocol followed in application layer for emails is Simple Mail Transfer Protocol (SMTP)

- What is the underlaying architecture and transport layer protocol used in e-mail application layer protocol?
    - The underlying architecture of the SMTP protocol is the Client – Server model, where Email clients communicate with email servers to send, receive and store emails. The Transport Layer protocol used in this instance is TCP

- Can you list down the basic steps involve in sending an e-mail from user A to B?
    - User A – Writes an email using an email client
    - The email client sends the email to the user's email server (SMTP server) using SMTP.
    - The email client establishes a connection with the SMTP server on port 25 and then the SMTP server authenticates User A after conluding the SMTP Handshake.
    - The SMTP server checks the domain of User B's email address so if User B's domain is different from User A's, the SMTP server looks up the destination domains Mail exchange record using DNS to find the receiver's mail server.
    - The SMTP server sends the email to User B's mail server.
    - User B's mail server receives and stores the email in User B's mailbox.
    - User B opens their email client, which connects to the mail server using either POP3 or IMAP (where the email is retrieved and stored).
    - User B reads the email.

Computer Networks and Communication
Task 3.1P DNS

Module Summary

A thorough summary of important networking ideas is given in SIT202: Computer networks and communication's "Learning Evidence for Active Class Task 3.1P". It goes over the Domain Name System (DNS), explaining how it works within the Application Layer of the TCP/IP paradigm and how it translates human-readable domain names into IP addresses. The significance of DNS and the need for a distributed system for redundancy and efficiency are explained in the paper. Additionally, it explains the HTTP request procedure, using role-playing to demonstrate the actions a browser takes to submit a request to a web server while highlighting proper protocol and communication. Furthermore covered is usage of nslookup and other tools for locating IP addresses and authoritative DNS servers, as well as a Wireshark analysis of DNS queries and answers. A description of email communication, with an emphasis on the Simple Mail Transfer Protocol (SMTP) and the procedures needed to send an email from one user to another, is provided in the document's conclusion.

Reflecting on the content

After giving this subject some thought, I now have a better knowledge of DNS and how important it is to internet connection. The thorough dissection of the HTTP request procedure helped me better understand client-server interactions by tying theoretical concepts to real-world situation. Practical expertise with network analysis was gained through the usage of Wireshark and nslookup, which is useful for real-world applications. Understanding SMTP and the intricate email communication process made it clear how complicated some common tasks might be which is directly related to comprehending and guaranteeing secure email transmission. The goal of the course team was to provide us a thorough understanding of fundamental networking protocols and technologies so that we could use them practically and perform efficient troubleshooting in real-world situations. This content enables us to effectively analyze and manage network communication by bridging the gap between theoretical knowledge and practical skills.

SIT202: Computer Networks and Communication
Leaning Evidence for Active Class Task 2

Name: Kenisha Corera
Student ID: C23020001

Members in this group activity task:

1. Nishad – C23110001
2. Kenisha – C23020001
3. Shekaina – C23110002
4. Raaid – C23020004
5. Pavithran – C22060015

Activity 1

1. Why do we need application layer protocols?
   - Applications need application layer protocols in order to communicate reliably, effectively, and securely over a network. Consequently, this improves the performance and usability of networked systems.
2. Discuss some examples of application layer protocols.
   - The World Wide Web's data communication system is based on the HTTP (Hypertext Transfer Protocol). From a web server to a web browser, it is used to transfer hypertext documents, such as HTML. In HTTP, requests are sent by clients, usually web browsers, to servers, who then provide the requested resource in return. Because there is no session data stored between queries, each request made using this protocol is stateless.

   - HTTPS (Hypertext transfer protocol secure) – HTTPS is a secure version of HTTP, used for safe communication between a web server and a browser. It combines HTTP with SSL/TLS to encrypt data, protecting sensitive information like passwords and payment details from interception. When using HTTPS, users see a padlock icon in their browser, indicating a secure connection. HTTPS also authenticates the server, ensuring users are communicating with the legitimate server.

   - File Transfer Protocol, or FTP, is a computer network protocol that facilitates file transfers between a client and a server. Users can upload, download, delete, rename, move, and copy files, and it gives them access to and control over files on a remote server..

   - Simple Mail Transfer Protocol (SMTP): SMTP is a network protocol used to transmit emails. Email clients can communicate with email servers and each other over this text-based protocol. To move email from the sender's email client to the recipient's email server, SMTP collaborates with the Mail Transfer Agent (MTA).

   - DNS (Domain name System) – DNS translates human-readable domain names (like www.example.com) into IP addresses that computers use to identify each other on the network. When a user enters a domain name into their browser, a DNS server translates this name into the

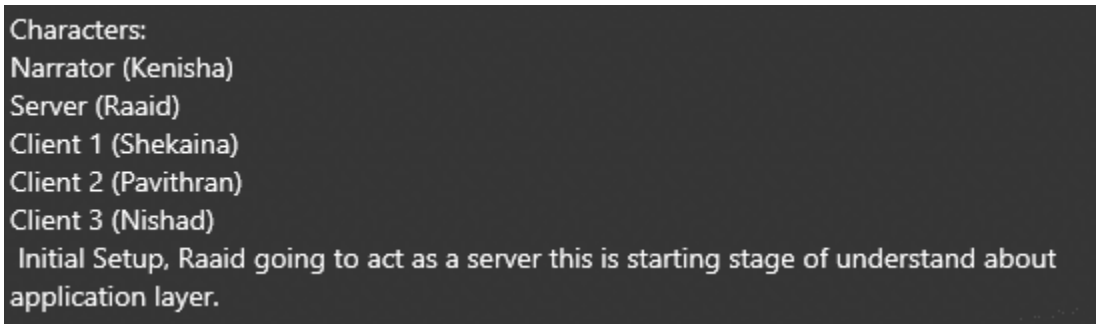
corresponding IP address, allowing the browser to locate and access the website. DNS is crucial for the usability of the internet, as it enables users to use easily memorable domain names instead of numerical IP addresses.

- SNMP, or Simple Network Management Protocol, is a tool for managing and keeping an eye on networks. Network performance management, problem detection and resolution, and network expansion planning are all made possible by it for administrators. The Management Information Base (MIB), an agent, and a manager make up SNMP. While the agents are the devices under the manager's control, the manager is in charge of the network. Details regarding the network devices' status are available in the MIB.

3. Let's consider one of the popular application layer protocols, HTTP to learn the principal operation of HTTP. To carry on this discussion, we can do a small role play.
    a. HTTP is a client-server protocol. Can you identify the key features of HTTP?

    ✓ The World Wide Web's cornerstone for data exchange is HTTP (Hypertext Transfer Protocol). Here are a few essential HTTP characteristics;
        - HTTP operates based on a request/response model where a client (such as a web browser) sends a request to a server, and the server responds with the requested resources.
        - HTTP is a stateless protocol, meaning each request from a client to a server is independent and does not require the server to retain session information or status between requests.
        - headers in HTTP provide essential information about the request or response, such as content type, content length, server details, client details.

    b. Identify a group member who could act as a web server and other three members can act as clients. So, your network has one server and three clients.



    c. Let's assume each client wants to access a web page stored in the web server and we use HTTP to communicate in the application layer level. You can now act as a server-client model communicating the right messages between each entity in sequence to get the information that each client wants. For example, Client 1 could say: "Client 1 initiating TCP connection with the Server". Then, the server replies with the correct response. You need to note down the correct messages in sequence.

Server: Alright team, let's start with the activity on understanding the application layer using HTTP. I'll be the web server, and the rest of you will be clients. Client 2, later you'll act as the proxy server. Ready?

Kenisha CICRA
Clients: Ready! Server and clients I'll introduce the clients, client 1 is Shekaina, Client 2 is pavithran and the last client 3 is Nishad

Shekaina CICRA
Client 1: Ok, I'm initiating TCP connection with the server

Server: Ok, I'm acknowledging TCP connection from Client 1. Connection established.

Shekaina CICRA
Client 1: I'm sending HTTP GET request for page1.html

Server: I'm responding with HTTP 200 OK and the content of page1.html.

Pavithran AIR
Client 2: Ok, I'm initiating TCP connection with the Server.

Server: Ok, I'm acknowledging TCP connection from Client 2. Connection established.

Pavithran AIR
Client 2: I'm sending HTTP GET request for page2.html.

Server: I'm responding with HTTP 200 OK and the content of page2.html.

Nishad
Client 3: Ok, I'm initiating TCP connection with the Server.

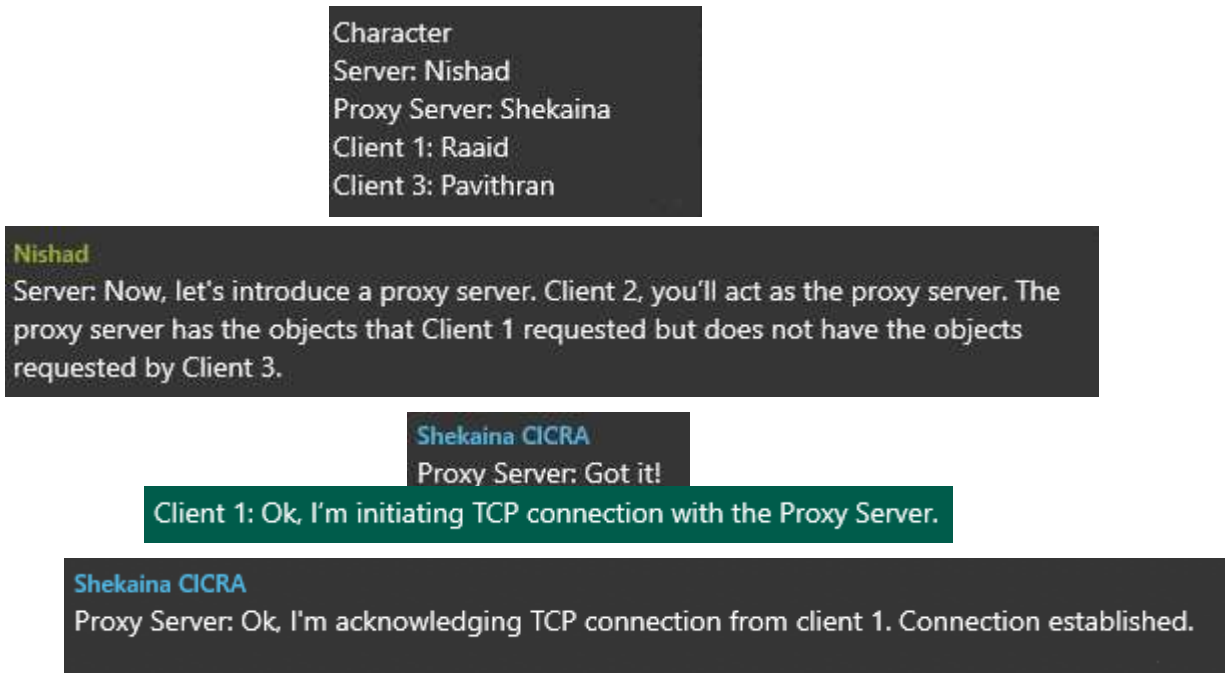
Server: Ok, I'm acknowledging TCP connection from Client 3. Connection established.

Nishad
Client 3: I'm sending HTTP GET request for page3.html.

Server: I'm responding with HTTP 200 OK and the content of page3.html.

d. You can maintain a timeline diagram as follows to show the messages that exchanged between the server and clients.

e. Once you have done the above steps (a to d), now assume a Web cache which is also called a proxy server introduced to the network as shown in the following diagram



Client 1: I'm sending HTTP GET request for page1.html.

Shekaina CICRA
Proxy Server: I'm responding with cached content of page1.html

Pavithran AIR
Client 3: Ok, I'm initiating TCP connection with the Proxy Server.

Shekaina CICRA
Proxy Server: Ok, I'm acknowledging TCP connection from client 3. Connection established.

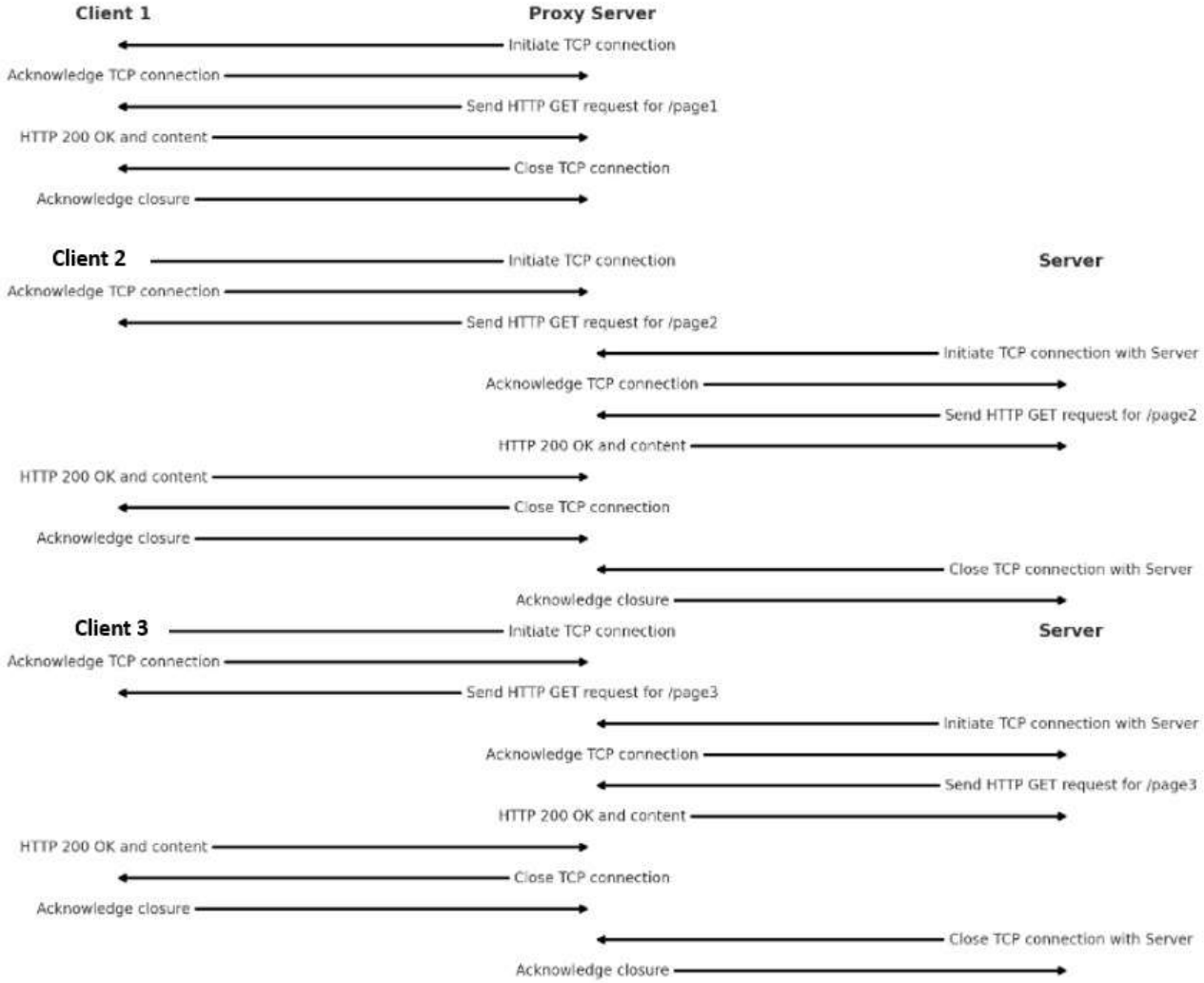
Pavithran AIR
Client 3: I'm sending HTTP GET request for page3.html.

Shekaina CICRA
Proxy Server: I'm forwarding the request to the original server.

Nishad
Server: Ok, I'm acknowledging TCP connection from the Proxy Server. Connection established. I'm responding with HTTP 200 OK and the content of page3.html to the

Shekaina CICRA
Proxy server(client 2): I'm forwarding the content of page3.html to client 3.

Kenisha CICRA
All note down the sequence of messages, ending point complete successfully

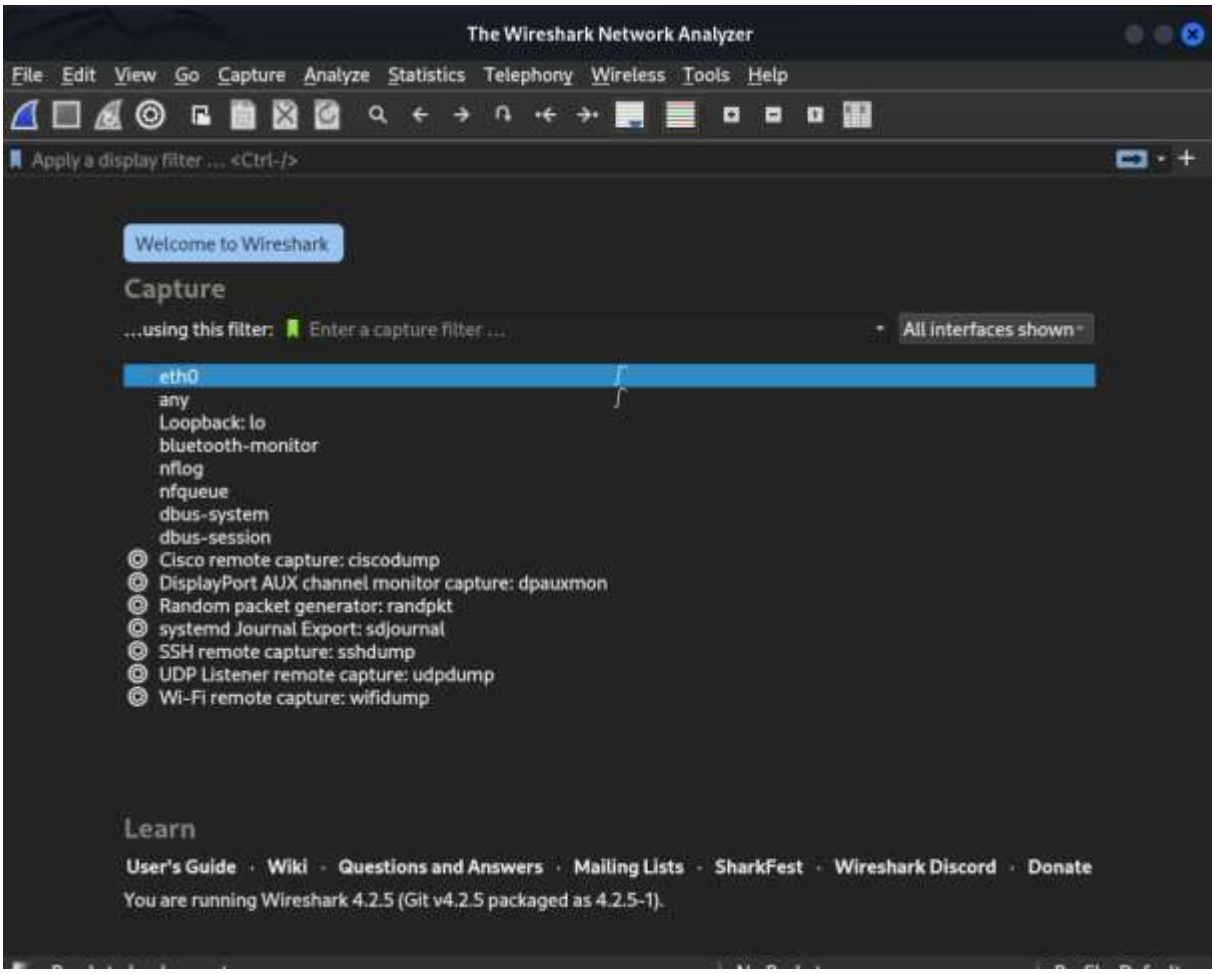

Client 1
Proxy Server
Initiate TCP connection
Acknowledge TCP connection
Send HTTP GET request for /page1
HTTP 200 OK and content
Close TCP connection
Acknowledge closure
Client 2
Initiate TCP connection
Server
Acknowledge TCP connection
Send HTTP GET request for /page2
Initiate TCP connection with Server
Acknowledge TCP connection
Send HTTP GET request for /page2
HTTP 200 OK and content
HTTP 200 OK and content
Close TCP connection
Acknowledge closure
Close TCP connection with Server
Acknowledge closure
Client 3
Initiate TCP connection
Server
Acknowledge TCP connection
Send HTTP GET request for /page3
Initiate TCP connection with Server
Acknowledge TCP connection
Send HTTP GET request for /page3
HTTP 200 OK and content
HTTP 200 OK and content
Close TCP connection
Acknowledge closure
Close TCP connection with Server
Acknowledge closure

Activity 2

a. Open your web browser and clear the browser's cache if you have not done that already

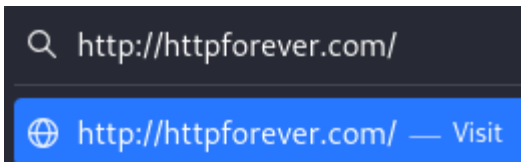
(Already completed this step)

b. Open the Wireshark packet sniffer and start the packet capture (Wireshark captures the packets by default when you open it for the first time)

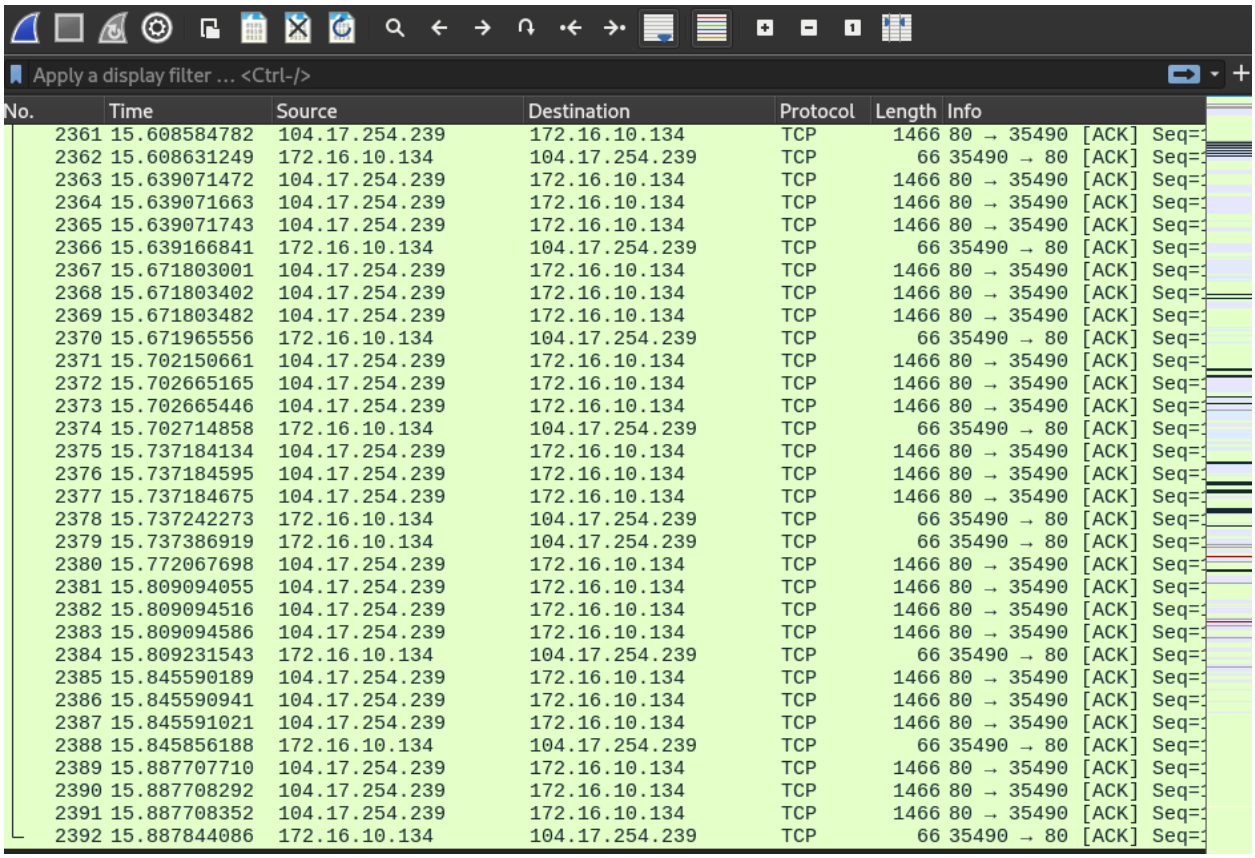


c. Enter the URL with http (not https) into your browser. If you are unsure what to use, you can use http://www.columbia.edu/~fdc/sample.html.

The webpage used for this activity is, http://httpforever.com/



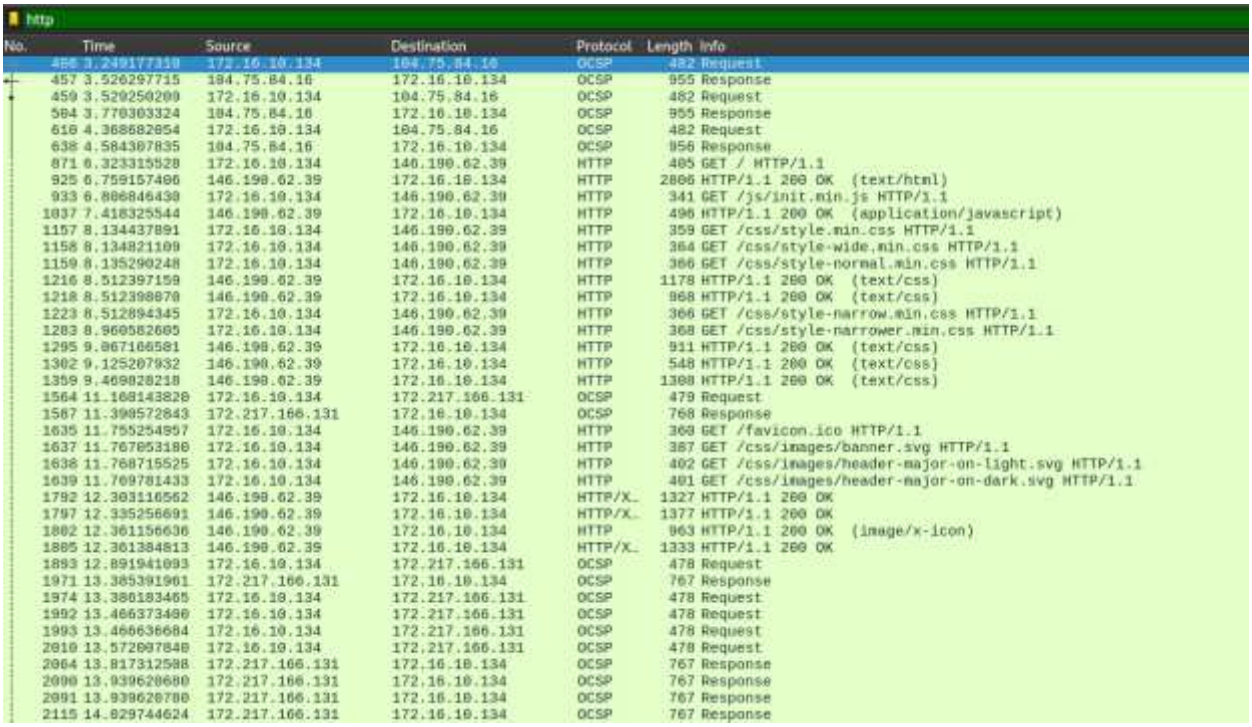
d. Stop Wireshark packet capture once the page is loaded in your browser.



Apply a display filter … <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
2361	15.608584782	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2362	15.608631249	172.16.10.134	104.17.254.239	TCP	66	35490 → 80 [ACK] Seq=1
2363	15.639071472	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2364	15.639071663	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2365	15.639071743	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2366	15.639166841	172.16.10.134	104.17.254.239	TCP	66	35490 → 80 [ACK] Seq=1
2367	15.671803001	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2368	15.671803402	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2369	15.671803482	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2370	15.671965556	172.16.10.134	104.17.254.239	TCP	66	35490 → 80 [ACK] Seq=1
2371	15.702150661	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2372	15.702665165	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2373	15.702665446	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2374	15.702714858	172.16.10.134	104.17.254.239	TCP	66	35490 → 80 [ACK] Seq=1
2375	15.737184134	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2376	15.737184595	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2377	15.737184675	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2378	15.737242273	172.16.10.134	104.17.254.239	TCP	66	35490 → 80 [ACK] Seq=1
2379	15.737386919	172.16.10.134	104.17.254.239	TCP	66	35490 → 80 [ACK] Seq=1
2380	15.772067698	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2381	15.809094055	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2382	15.809094516	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2383	15.809094586	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2384	15.809231543	172.16.10.134	104.17.254.239	TCP	66	35490 → 80 [ACK] Seq=1
2385	15.845590189	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2386	15.845590941	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2387	15.845591021	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2388	15.845856188	172.16.10.134	104.17.254.239	TCP	66	35490 → 80 [ACK] Seq=1
2389	15.887707710	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2390	15.887708292	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2391	15.887708352	104.17.254.239	172.16.10.134	TCP	1466	80 → 35490 [ACK] Seq=1
2392	15.887844086	172.16.10.134	104.17.254.239	TCP	66	35490 → 80 [ACK] Seq=1

e. We want to analyse HTTP. Therefore, you want to filter out “http” responses. To do that, enter “http” in the filter.



http

No.	Time	Source	Destination	Protocol	Length	Info
455	3.249177319	172.16.10.134	104.75.84.16	OCSP	482	Request
457	3.526297715	104.75.84.16	172.16.10.134	OCSP	955	Response
459	3.529250209	172.16.10.134	104.75.84.16	OCSP	482	Request
504	3.770303324	104.75.84.16	172.16.10.134	OCSP	955	Response
616	4.368882054	172.16.10.134	104.75.84.16	OCSP	482	Request
638	4.584307835	104.75.84.16	172.16.10.134	OCSP	956	Response
871	6.323315528	172.16.10.134	146.190.62.39	HTTP	405	GET / HTTP/1.1
925	6.759157496	146.190.62.39	172.16.10.134	HTTP	2806	HTTP/1.1 200 OK (text/html)
933	6.806846439	172.16.10.134	146.190.62.39	HTTP	341	GET /js/init.min.js HTTP/1.1
1037	7.418325544	146.190.62.39	172.16.10.134	HTTP	496	HTTP/1.1 200 OK (application/javascript)
1157	8.134437891	172.16.10.134	146.190.62.39	HTTP	359	GET /css/style.min.css HTTP/1.1
1158	8.134821109	172.16.10.134	146.190.62.39	HTTP	364	GET /css/style-wide.min.css HTTP/1.1
1159	8.135290248	172.16.10.134	146.190.62.39	HTTP	366	GET /css/style-normal.min.css HTTP/1.1
1216	8.512397159	146.190.62.39	172.16.10.134	HTTP	1178	HTTP/1.1 200 OK (text/css)
1218	8.512398870	146.190.62.39	172.16.10.134	HTTP	968	HTTP/1.1 200 OK (text/css)
1223	8.512894345	172.16.10.134	146.190.62.39	HTTP	366	GET /css/style-narrow.min.css HTTP/1.1
1283	8.960582605	172.16.10.134	146.190.62.39	HTTP	368	GET /css/style-narrower.min.css HTTP/1.1
1295	9.067166581	146.190.62.39	172.16.10.134	HTTP	911	HTTP/1.1 200 OK (text/css)
1302	9.125207932	146.190.62.39	172.16.10.134	HTTP	548	HTTP/1.1 200 OK (text/css)
1359	9.469028218	146.190.62.39	172.16.10.134	HTTP	1308	HTTP/1.1 200 OK (text/css)
1564	11.168143828	172.16.10.134	172.217.166.131	OCSP	479	Request
1567	11.398572843	172.217.166.131	172.16.10.134	OCSP	768	Response
1635	11.755254957	172.16.10.134	146.190.62.39	HTTP	360	GET /favicon.ico HTTP/1.1
1637	11.767053180	172.16.10.134	146.190.62.39	HTTP	387	GET /css/images/banner.svg HTTP/1.1
1638	11.768715525	172.16.10.134	146.190.62.39	HTTP	402	GET /css/images/header-major-on-light.svg HTTP/1.1
1639	11.769781433	172.16.10.134	146.190.62.39	HTTP	401	GET /css/images/header-major-on-dark.svg HTTP/1.1
1792	12.303116562	146.190.62.39	172.16.10.134	HTTP/X…	1327	HTTP/1.1 200 OK
1797	12.335256691	146.190.62.39	172.16.10.134	HTTP/X…	1377	HTTP/1.1 200 OK
1802	12.361156636	146.190.62.39	172.16.10.134	HTTP	963	HTTP/1.1 200 OK (image/x-icon)
1805	12.361384813	146.190.62.39	172.16.10.134	HTTP/X…	1333	HTTP/1.1 200 OK
1893	12.891941093	172.16.10.134	172.217.166.131	OCSP	478	Request
1971	13.385391961	172.217.166.131	172.16.10.134	OCSP	767	Response
1974	13.386183465	172.16.10.134	172.217.166.131	OCSP	478	Request
1992	13.466373480	172.16.10.134	172.217.166.131	OCSP	478	Request
1993	13.466636684	172.16.10.134	172.217.166.131	OCSP	478	Request
2010	13.572087848	172.16.10.134	172.217.166.131	OCSP	478	Request
2064	13.817312588	172.217.166.131	172.16.10.134	OCSP	767	Response
2090	13.939620680	172.217.166.131	172.16.10.134	OCSP	767	Response
2091	13.939620780	172.217.166.131	172.16.10.134	OCSP	767	Response
2115	14.029744624	172.217.166.131	172.16.10.134	OCSP	767	Response

f. Now you can analyse HTTP, requests, responses, and sequences.

(Analyzed and answered in questions below)

g. By analysing the HTTP responses, you should be able to answer the following questions,

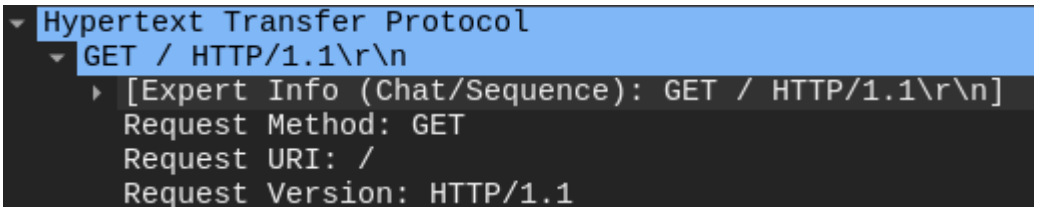
a. What is the sequence of HTTP message exchange?

HTTP/1.1 200 OK (text/html)
GET /js/init.min.js HTTP/1.1
HTTP/1.1 200 OK (application/javascript)
GET /css/style.min.css HTTP/1.1
GET /css/style-wide.min.css HTTP/1.1
GET /css/style-normal.min.css HTTP/1.1
HTTP/1.1 200 OK (text/css)
HTTP/1.1 200 OK (text/css)
GET /css/style-narrow.min.css HTTP/1.1
HTTP/1.1 200 OK (text/css)
HTTP/1.1 200 OK (text/css)
HTTP/1.1 200 OK (text/css)
Request
Response
GET /favicon.ico HTTP/1,1
GET /css/images/banner,svg HTTP/1.1
GET /css/images.header-major-on-light.svg HTTP/1.1
GET /css/images/header-major-on-dark.svg HTTP/1.1
HTTP/1.1 200 OK
HTTP/1.1 200 OK
HTTP/1.1 200 OK (image/x-icon)
HTTP/1,1 200 OK

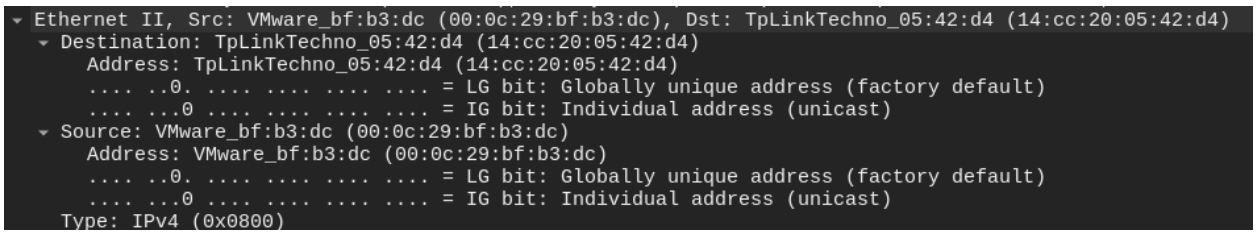
b. Are we using persistence/ non-persistence connection?

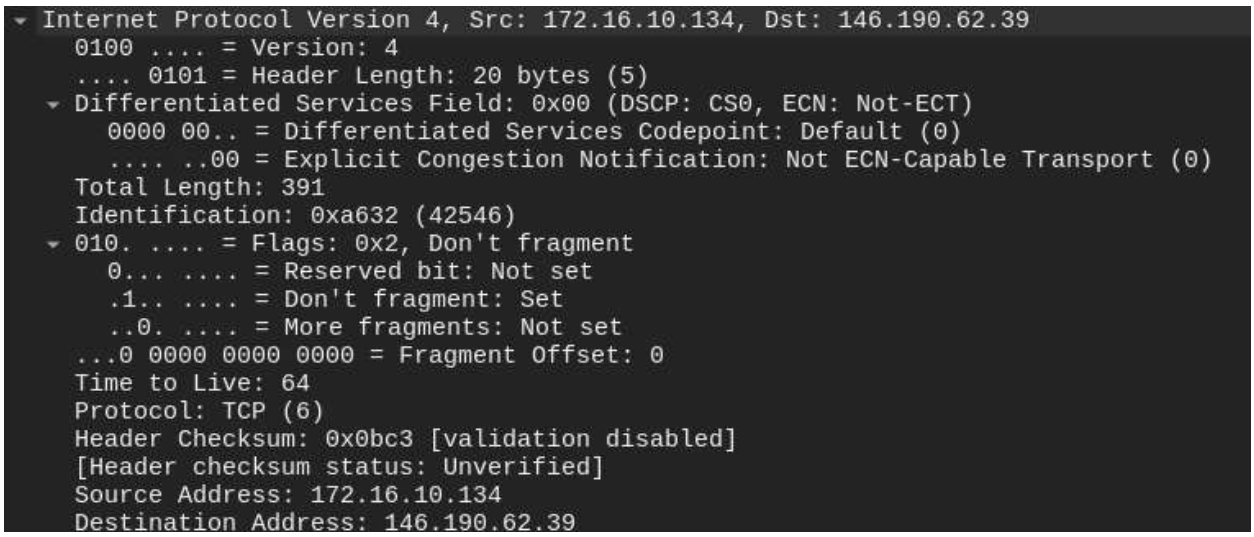
This website is using persistence connection because there are Multiple Requests and Responses between source and destination addresses. Which means that the connection is being reused which is prevalent in persistent connections.

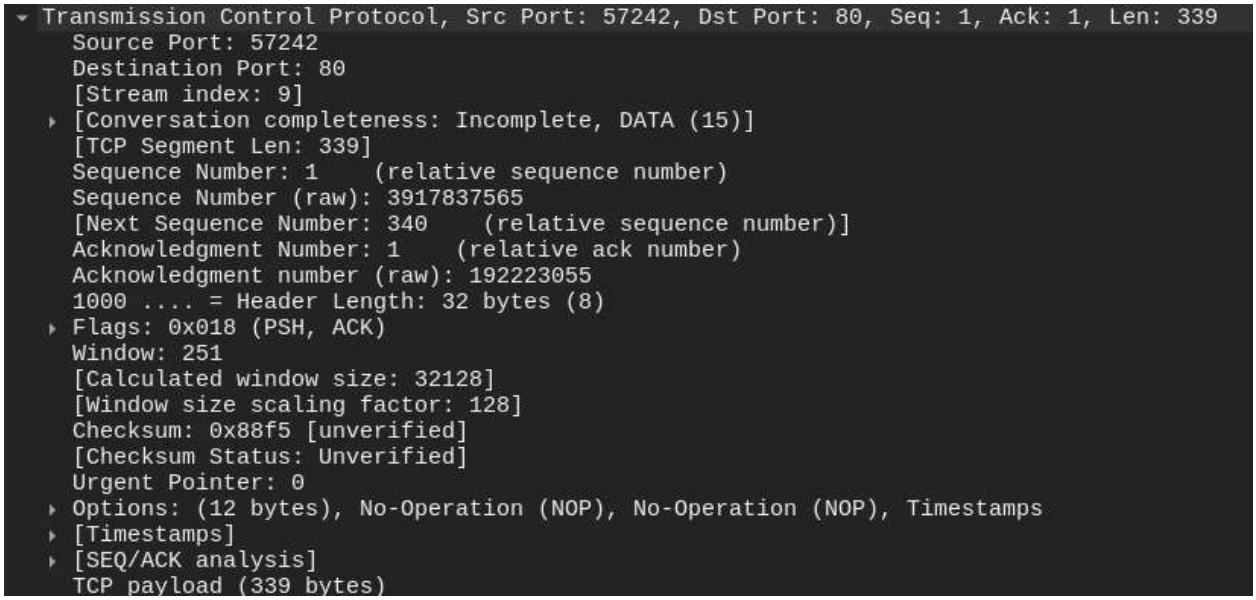
c. What are details you can find in HTTP GET message?

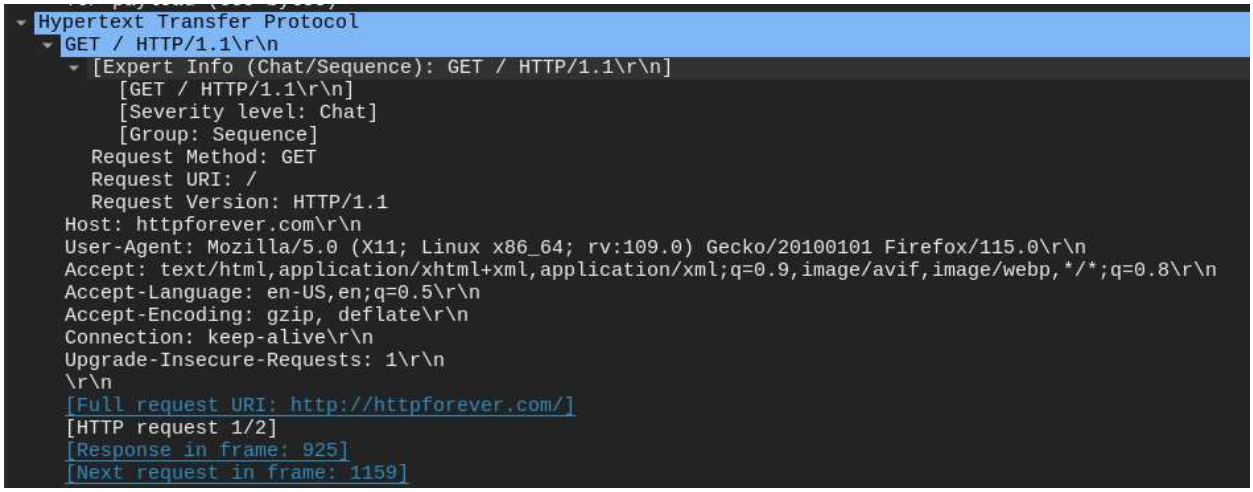


d. Check the packet details in the middle Wireshark packet details pane. Can you identify the details in Ethernet II / Internet Protocol Version 4 / Transmission Control Protocol / Hypertext Transfer Protocol frames?









e. What is the HTTP version your browser used?

- Version 1.1 of HTTP is in use.

f. Identify the response message received from the server?

- All was well with the message from the server.

g. What is version of HTTP that the server is running?

- The server is running HTTP version 1.1.

h. Can you identify the IP address of your computer?

- 172.16.10.134

i. Can you identify the IP address of http://www.columbia.edu/ server?

- Given that we utilized the httpforever website, the webserver's IP address is 146.190.62.39.

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

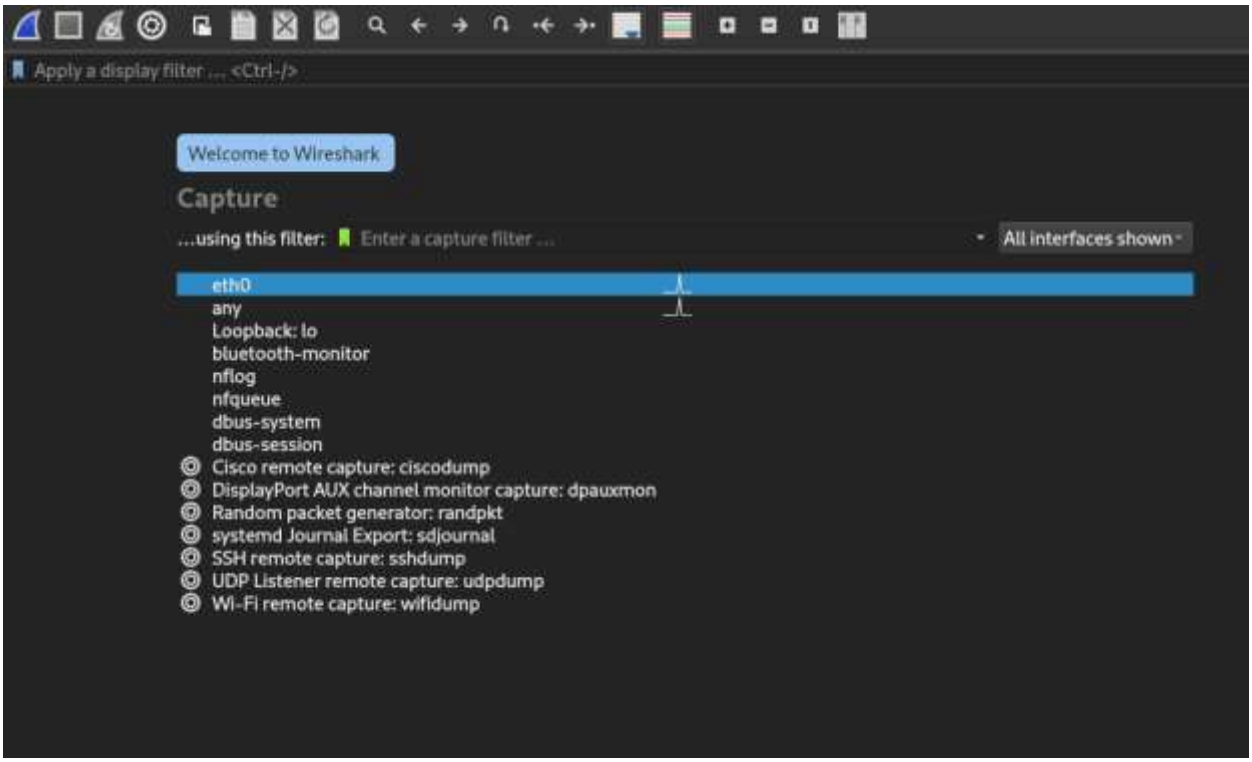
- The server has sent the browser a status code of 200 (OK).

k. When was the HTML file that you are accessing last modified at the server?

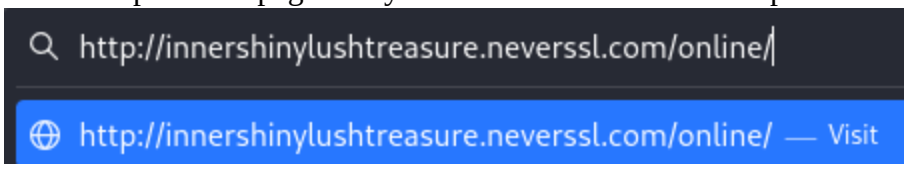
- The last update to the HTML file was made on November 16, 2022.

l. Repeat the analysis by accessing a different web page of your choice.

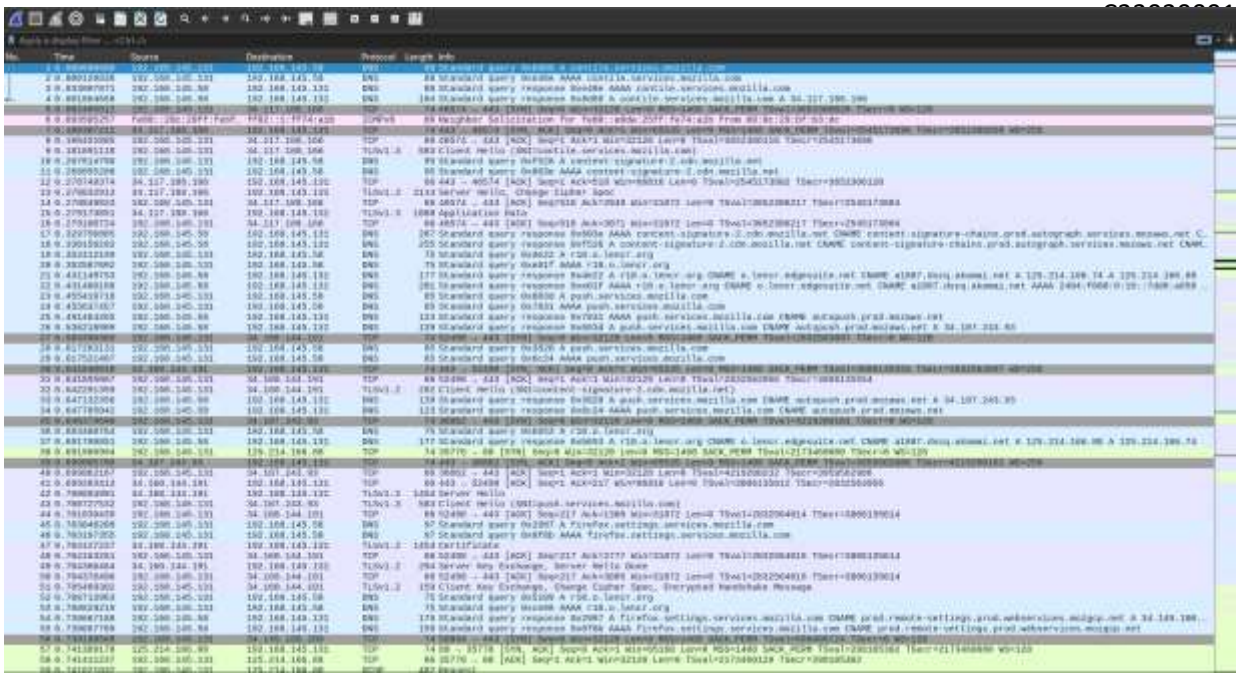
The HTTP webpage I'll be analyzing for this question is http://innershinylushtreasure.neverssl.com/online/
I shall begin by first open wireshark and begin capturing packets.



I shall now load up the webpage in my browser so wireshark can capture traffic.



Once the website has loaded, I shall end packet capturing and filter results by 'http' (as seen in the next page).



http

No.	Time	Source	Destination	Protocol	Length	Info
59	0.741621032	192.168.145.131	125.214.166.88	OCSP	482	Request
70	0.800815334	125.214.166.88	192.168.145.131	OCSP	956	Response
74	0.810319218	192.168.145.131	125.214.166.88	OCSP	482	Request
88	0.892633881	125.214.166.88	192.168.145.131	OCSP	955	Response
89	0.892944416	192.168.145.131	125.214.166.88	OCSP	482	Request
103	0.947012983	125.214.166.88	192.168.145.131	OCSP	955	Response
122	1.001038552	192.168.145.131	125.214.166.88	OCSP	482	Request
126	1.047066482	125.214.166.88	192.168.145.131	OCSP	956	Response
213	4.584089429	192.168.145.131	34.223.124.45	HTTP	432	GET /online/ HTTP/1.1
216	5.202572631	34.223.124.45	192.168.145.131	HTTP	1612	HTTP/1.1 200 OK (text/html)
222	6.119178326	192.168.145.131	34.223.124.45	HTTP	407	GET /favicon.ico HTTP/1.1
224	6.428223348	34.223.124.45	192.168.145.131	HTTP	509	HTTP/1.1 200 OK (PNG)

As you can see above. The communication between server and client has been captured just like in the previous instance.

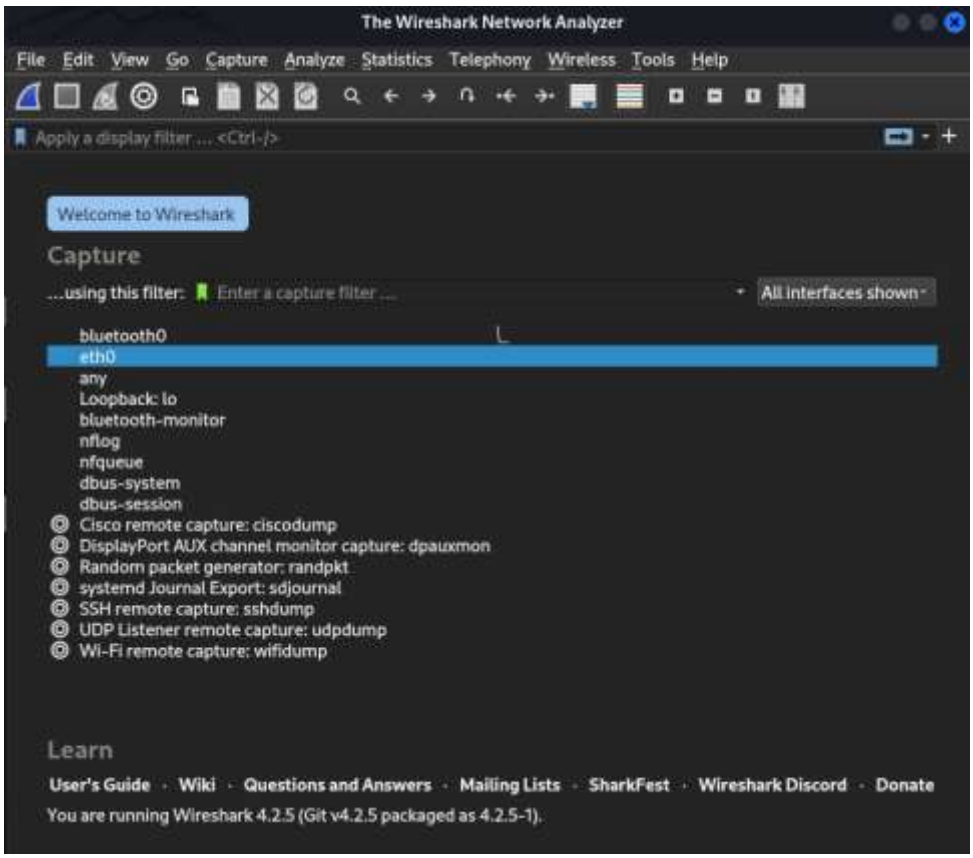
The sequence of HTTP message exchange in this instance is as follows,

- GET /online/ HTTP/1.1
- HTTP/1.1 200 OK (text/html)
- GET /favicon.ico HTTP/1.1
- HTTP/1.1 200 OK (PNG)

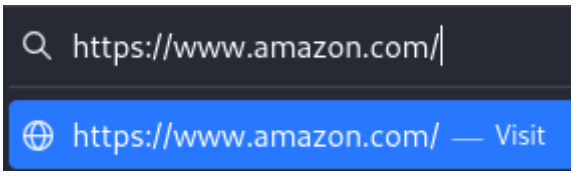
We can also see that both browser and server use HTTP version 1.1 and the status code returned from the server to the browser is 200 (OK). The Client IP address is 192.168.145.131 and the Server IP address is 34.223.124.45 and finally the last modified date of the HTML file was on 29th June 2022

h. What happens when you use “https”? Can you analyse the responses? Discuss your answer with your group members.

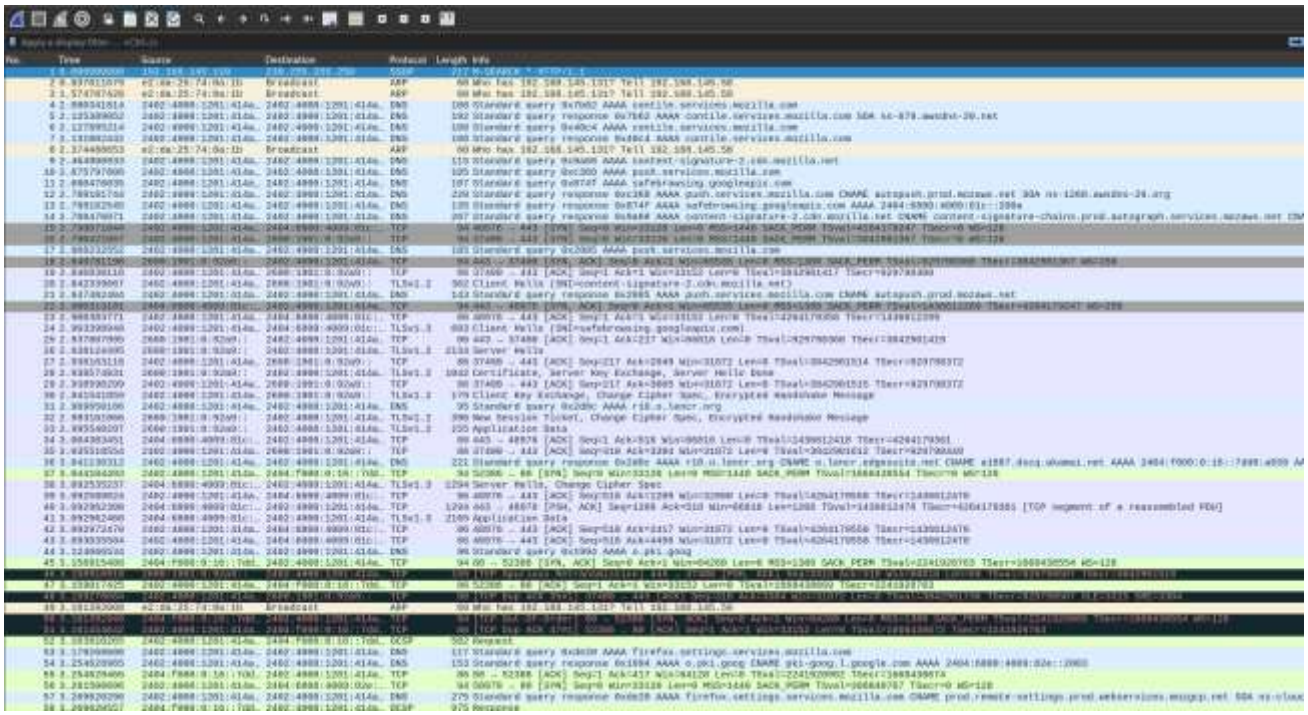
- To begin, we open Wireshark and begin capturing packets as instructed.



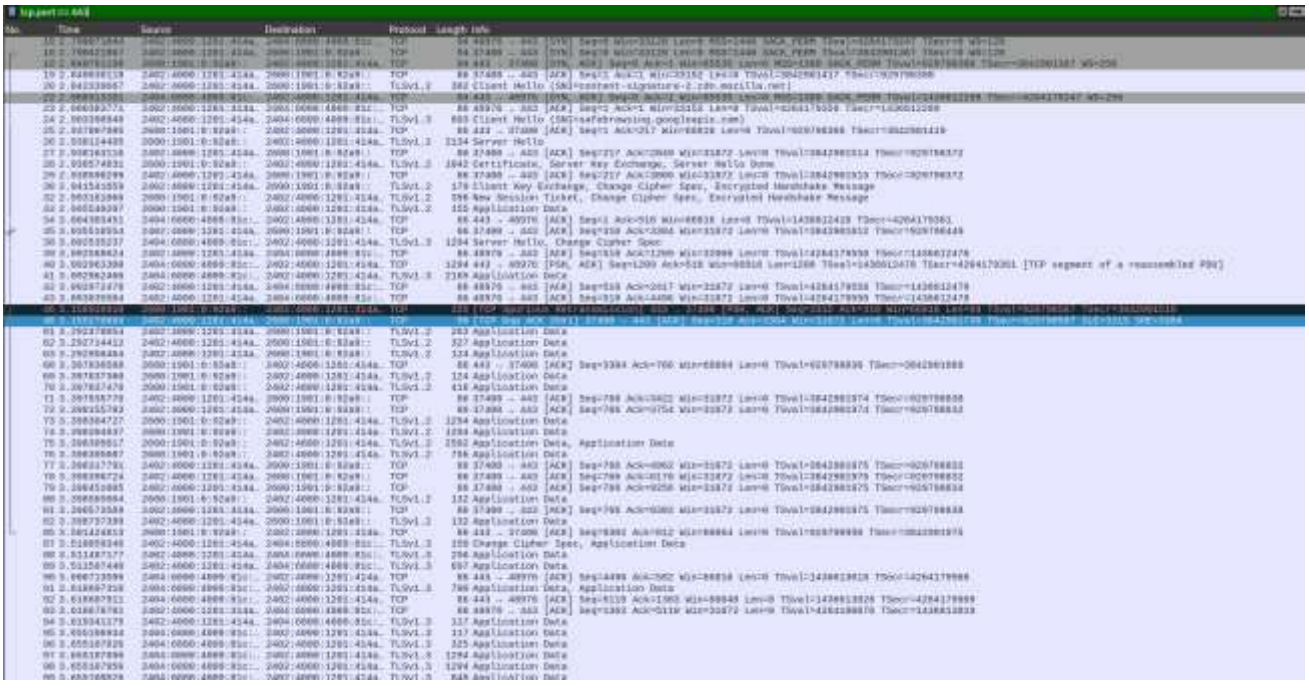
- Then we load up the website on the browser. For this example, we chose https://www.amazon.com/



- Once the website has loaded successfully, we end packet capturing.



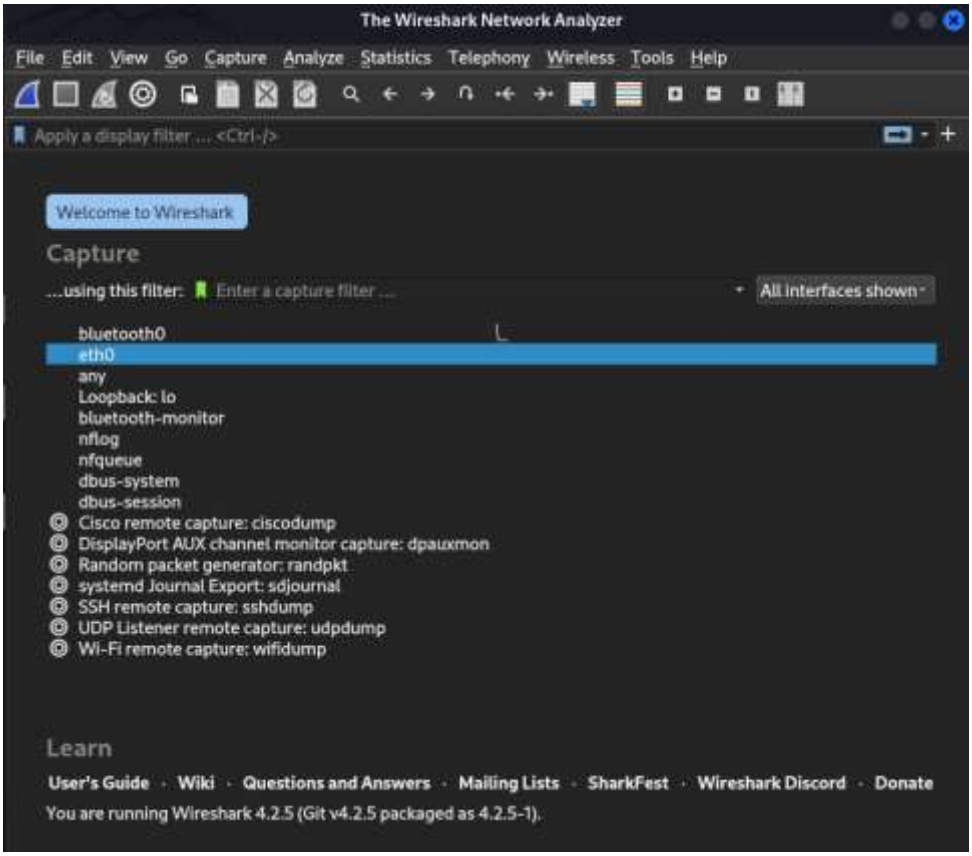
- Since HTTPS uses TCP port 443 we can filter HTTPS traffic by using the filter ‘tcp.port == 443’



- As mentioned in the Above and Beyond Task. Wireshark cannot analyze HTTPS due to its encryption. So the only possible method of displaying HTTPS traffic is by decrypting it using SSL and TLS packets.

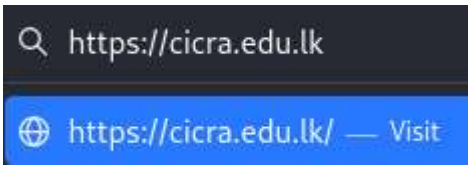
Above and Beyond Tasks

1. Open your web browser and clear the browser's cache. Open the Wireshark packet sniffer and start the packet capture

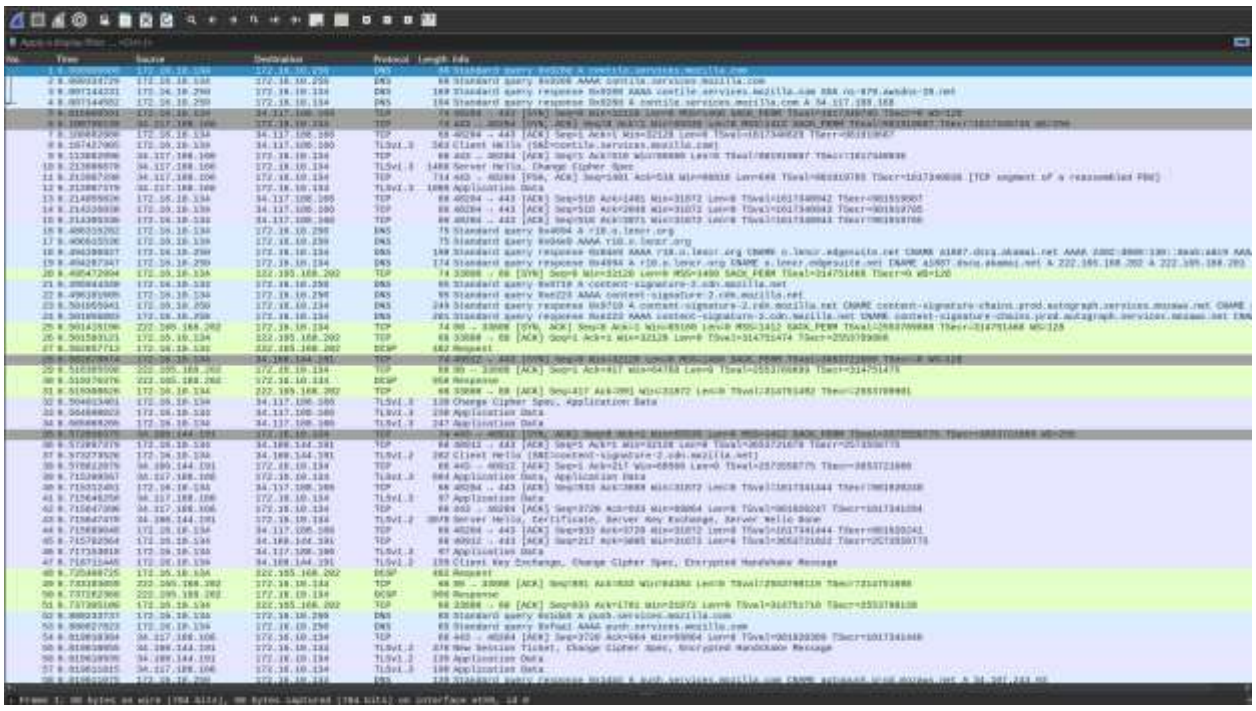


2. Enter a URL with HTTPS in your browser.

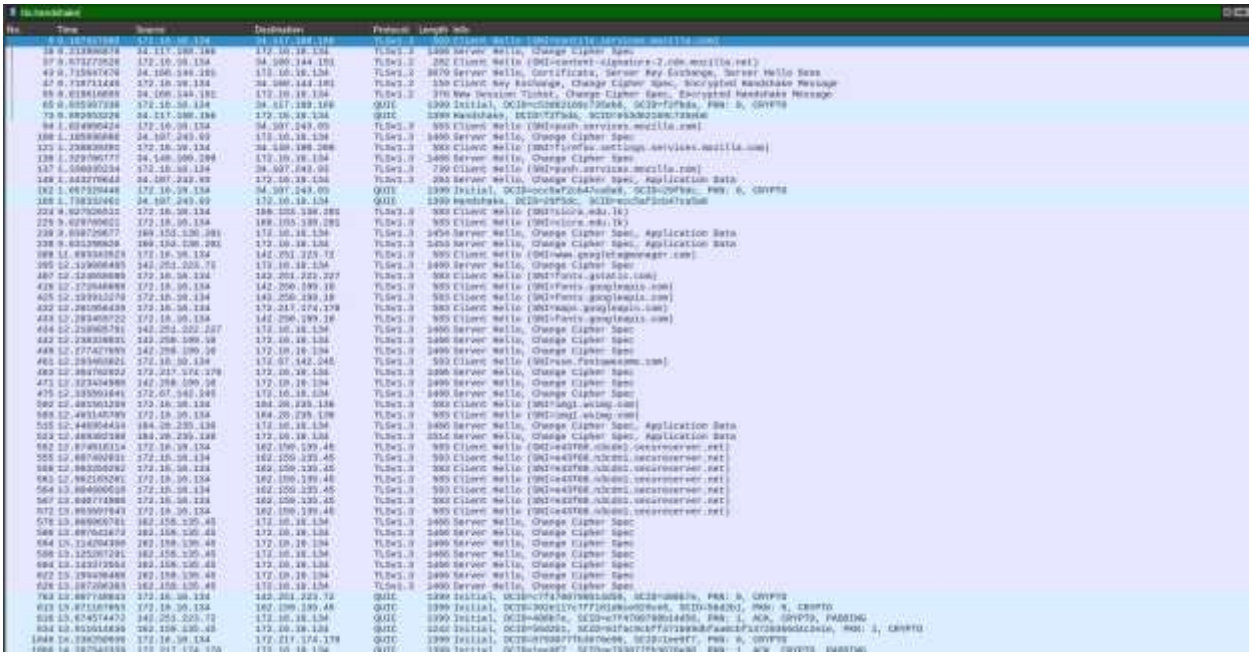
   The webpage used for this activity is, https://cicra.edu.lk/



3. Stop packet capture and you can start analysing the packets. Explain the operation and handshake process of TLS using the screen captures of Wireshark.



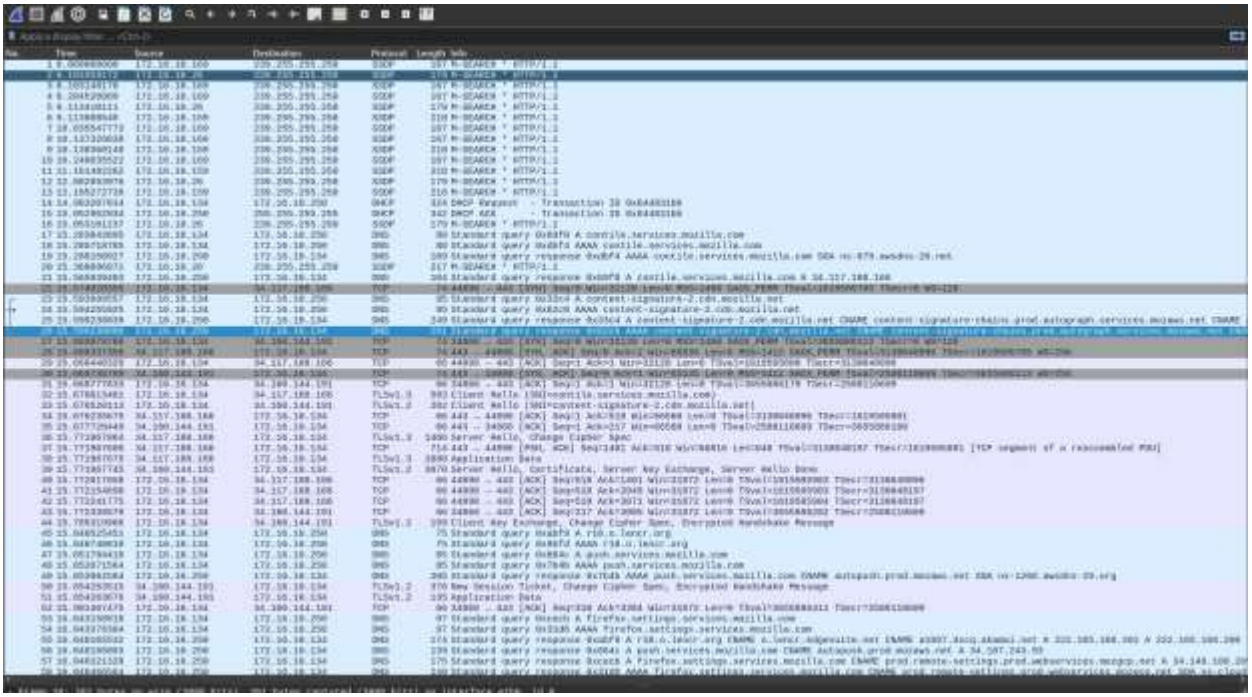
4. You need to clearly identify the message sequence and protocols used (including transport layer protocols) before your browser sends the first HTTP GET message to the relevant web server.



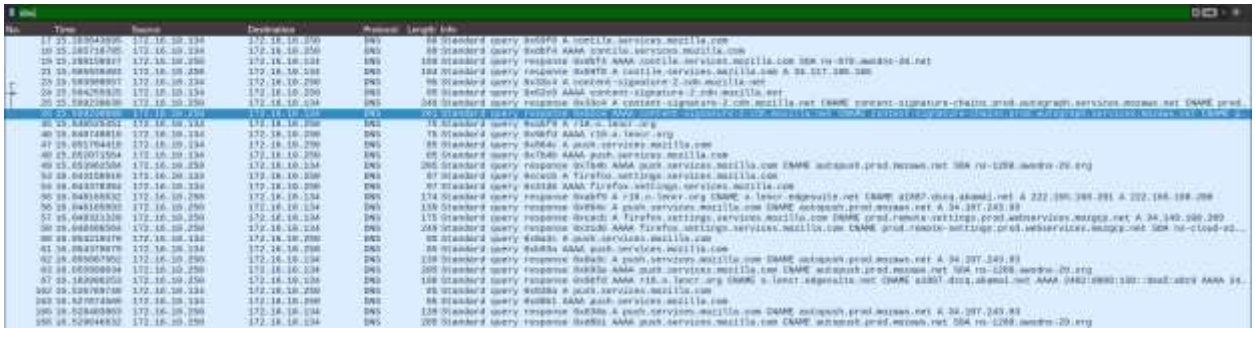
As you can see in the previous page by filtering with the keyword “tls.handshake” we can witness the entire handshake process of TLS. Its sequence is as follows,

- Client Hello
- Server Hello
- Certificate
- Server Key Exchange
- Server Hello Done
- Client Key Exchange
- Change Cipher Spec
- Encrypted Handshake Message
- Handshake

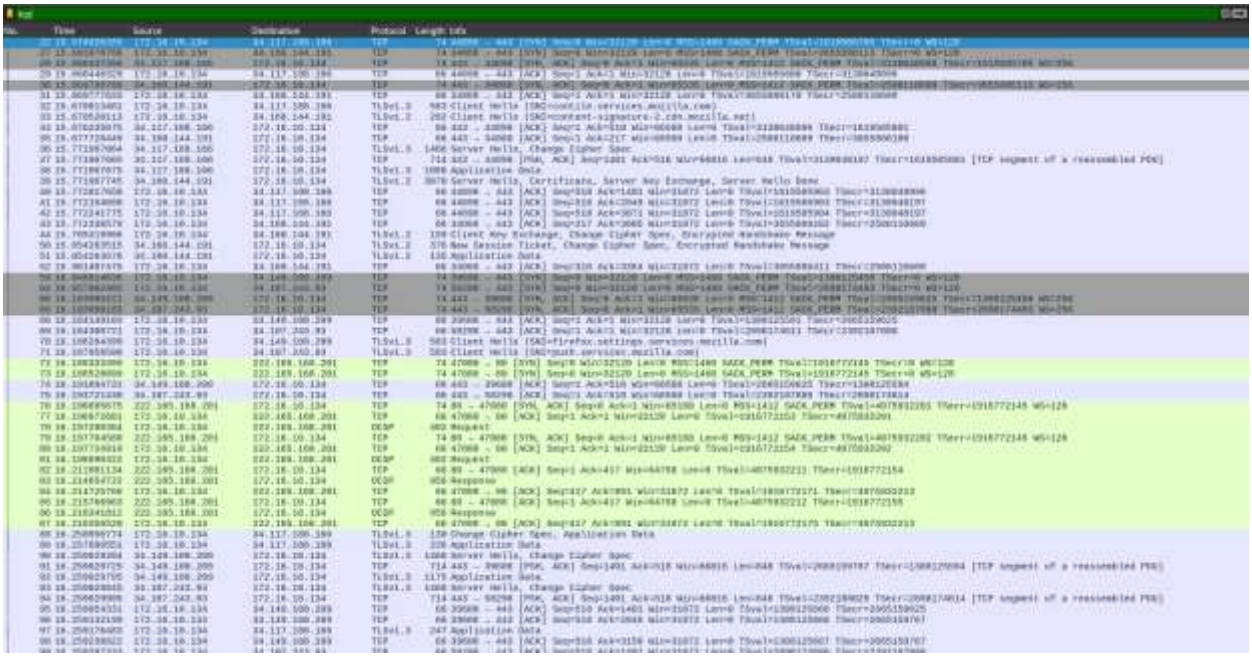
Below are the results from traffic captured before opening a link in a web browser.



If we filter by DNS protocol,



We can see the query and response packets between the browser and the web server.



By filtering using the TCP protocol as seen above, we can notice the three-way handshake, which establishes a TCP connection between your browser and the web server.

We can see the ‘SYN’ (which is sent from the client browser to initiate a connection), ‘SYN-ACK’ (which is sent from the server to confirm the ‘SYN’ packet and finally the ‘ACK’ sent from the client confirming the ‘SYN-Ack’ packet.

5. Can you analyse HTTPS in Wireshark? Explain your answer. If yes, provide evidence on how we can do that. If not, is there any alternative method we could use to analyse HTTPS?

By default, Wireshark cannot analyse HTTPS due to its encryption but you can display the SSL and TLS packets and decrypt them to monitor HTTPS traffic.

SIT202 – Computer Networks and Communication
Task 3.1 P

Module Summary

- We studied application layer protocols in this topic, which are essential to network communication. Important protocols covered in the conversation include: We studied application layer protocols in this topic, which are essential to network communication. Important protocols covered in the conversation include:

    1. Hypertext Transfer Protocol
        - With HTTP, hypertext documents can be transferred between web servers and browsers, which is crucial for the functioning of the internet. A request/response paradigm underpins this stateless protocol.

    2. Hypertext Transfer Protocol Secure
        - With the addition of SSL/TLS to encrypt data, HTTPS improves HTTP by guaranteeing secure communication and safeguarding private information.

    3. File Transfer Protocol
        - Users can upload, download, and manage data on distant servers with the help of File Transfer Protocol (FTP), which speeds up file transfers between clients and servers.

    4. Simple Mail Transfer protocol
        - Email clients can connect with servers and with one another thanks to SMTP, which is used for email transfer.

    5. Domain Name System
        - Websites can be accessed with easily remembered names rather than numeric IP addresses thanks to DNS, which converts domain names into IP addresses.

    6. Simple Network Management Protocol
        - Network administrators can control network performance, identify issues, and make expansion plans for their networks by using SNMP for monitoring and management of networks.

- We used Wireshark to record and examine HTTP and HTTPS traffic, and we role-played HTTP interactions between clients and a server in order to gain a deeper understanding of these protocols.

Reflection Content

1. Most important thing that I learn

- Especially useful was the thorough comprehension of HTTP and HTTPS. Understanding how clients and servers request and transfer data, as well as how HTTPS secures this communication, was essential. For web development and cybersecurity, this information is fundamental.

- My knowledge of how websites operate was rudimentary before this module. My understanding of online communication was expanded by this module, which outlined the fundamental protocols. It made a great connection with my previous web development and fundamental networking courses.

- An understanding of application layer protocols is essential for anyone working in networking or information technology, and this was the goal of the course team. They wanted to make sure we could apply theory to real-world situations, so they got us involved in hands-on activities like packet analysis and role-playing.

- Anyone looking to work in IT can benefit greatly from the knowledge acquired, especially in positions involving web development, networking, and cybersecurity. Comprehending these protocols is crucial for the creation, administration, and protection of networked systems.

- Practical experience was gained through role-playing HTTP conversation and using Wireshark to analyze packet data. My comprehension of the functioning of these protocols and their importance in network communication has been strengthened by this real-world application.

- To summarize, this module provided a thorough review of fundamental application layer protocols together with hands-on exercises that strengthened my comprehension and equipped me for practical applications in the communication and computer network fields.

SIT202: Computer Networks and Communication

Leaning Evidence for Active Class Task 4

Name: Kenisha Corera
Student ID: C23020001

Members in this group activity task:
Nishad – C23110001
Kenisha – C23020001
Shekaina – C23110002
Raaid – C23020004
Pavithran – C22060015

Activity 1

1. Group discussion: Assume that you are helping your friend to move his/her house and your responsibility is to take care of moving goods in the kitchen.
    a. What method would you take if your friend needs to pack everything within an hour because the moving truck will be there soon?
        - If your friend has to pack everything in an hour since the moving truck is coming shortly, then efficiency and speed should be the main priorities. Here's a methodical approach:
            - Set Essential Items in Order of Priority: Pack the most important things first, such as dishes, cutlery and regularly used equipment.
            - Make use of the Containers available: Don't stress too much about sorting just use any cartons, bags or containers that are available.
            - Gather Fastly: As soon as you can, stuff everything into boxes without giving too much thought to organize.
            - Reduce the amount of Wrapping: To save time, only lightly wrap delicate goods.

    b. How would your method change if you have more time and your friend would like to make sure each item in the kitchen is safely delivered, and your friend wants to keep those on the same order it was on the previous house?
        - A strategy centered on meticulous packing and organizing would be used if you had more time and your time and your friend wanted to make sure every kitchen item was delivered securely and maintained in the same order as in the previous house:
            - Sort Items into Categories: Put utensils, dishes, pots and pans and appliance in group with related goods.
            - Marker Boxes: Indicate on each box exactly what's inside and which room in belongs in.
            - Use appropriate packaging supplies: To safeguard delicate things use robust boxes, packing paper and bubble wrap.
            - Keep the Peace: Assemble the object in the boxes according to their original arrangement in the kitchen by packing them logically.
            - Make a list: Make an inventory list before the relocation to ensure nothing is forgotten or lots

c. Now, can you relate the above scenarios to the well-known two transport layer protocols? Consider different scenarios including large file transfer and faster communication.

- In networking, the two most well-known transport layer protocols are UDP (user Datagram Protocol) and TCP (Transmission Control Protocols). Here's how we can connect the aforementioned scenarios to these protocols:
- Scenario 1- Fast Packing (Resembling UDP)
  - UDP (User Datagram Protocol): Instead of reliability, this protocol prioritizes speed. It transmits packets in an unconnected manner and makes no assurances on delivery, sequencing or error-checking.
  - In connection with Fast packing: It is similar to UDP in that everything gets packed in less than an hour. Moving everything quickly is the main objective, even if it means that certain things may not be packed precisely or may get mixed up. It's speed not dependability that matters.
- Scenario 2- Thorough Packing (Comparable to TCP)
  - TCP (Transmission Control Protocol): This protocol guarantees packet delivery that is dependable well- organized and error- checked. Prior to data transfer it creates a link and preserves packet order
  - Connection to Cautionary Packing: An approach that is similar to TCP involves taking extra time to pack, guaranteeing that every item is delivered safely and maintained in the same sequence. Making ensuring everything is properly packaged and delivered in the right sequence is the main objective. Even if it takes longer, accuracy and reliability come first.
- Different Scenarios:
  Fast Communication (Similar to UDP)
  - In application such as online gaming and live broadcasting speed is more important than reliability. Since minor packet losses may be tolerated, UDP is a preferable option because of its quicker transmission speed and reduced latency.

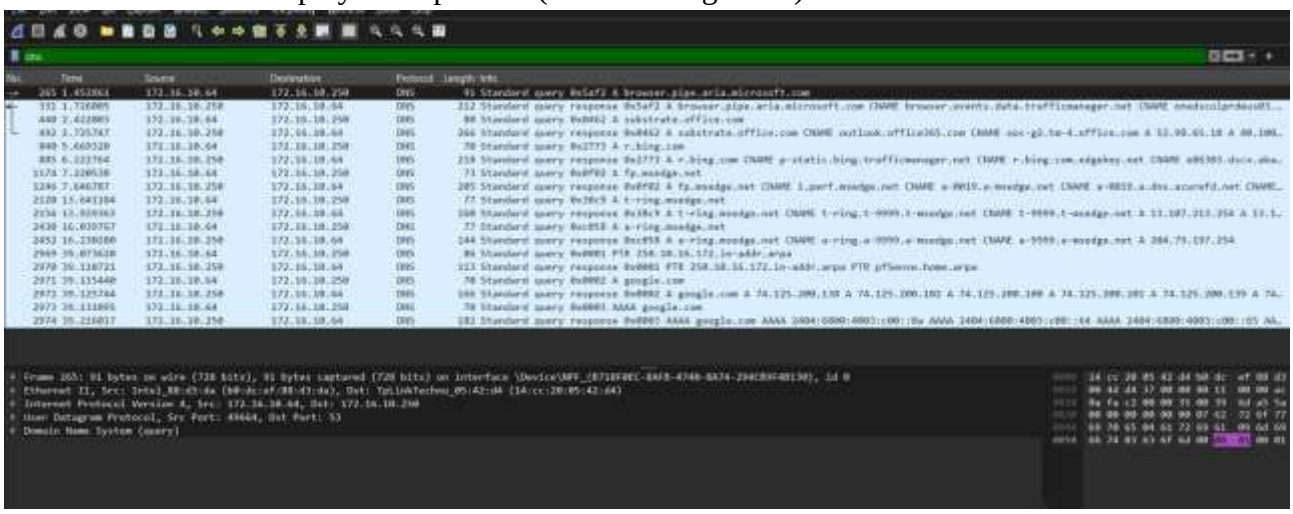
  Large File Transfer (Similar to TCP)
  - To guarantee that the complete file is received accurately while transferring huge files, dependability and order are essential. Since TCP guarantees data integrity and proper sequencing it is perfect for this use case.

To summarize, careless packing is similar to TCP in that it ensures order and dependability at the expense of speed whereas rapid packing is similar to UDP in that it prioritize speed above reliability. Which technique or protocol is best will depend on the many applications or scenarios.

Activity 2

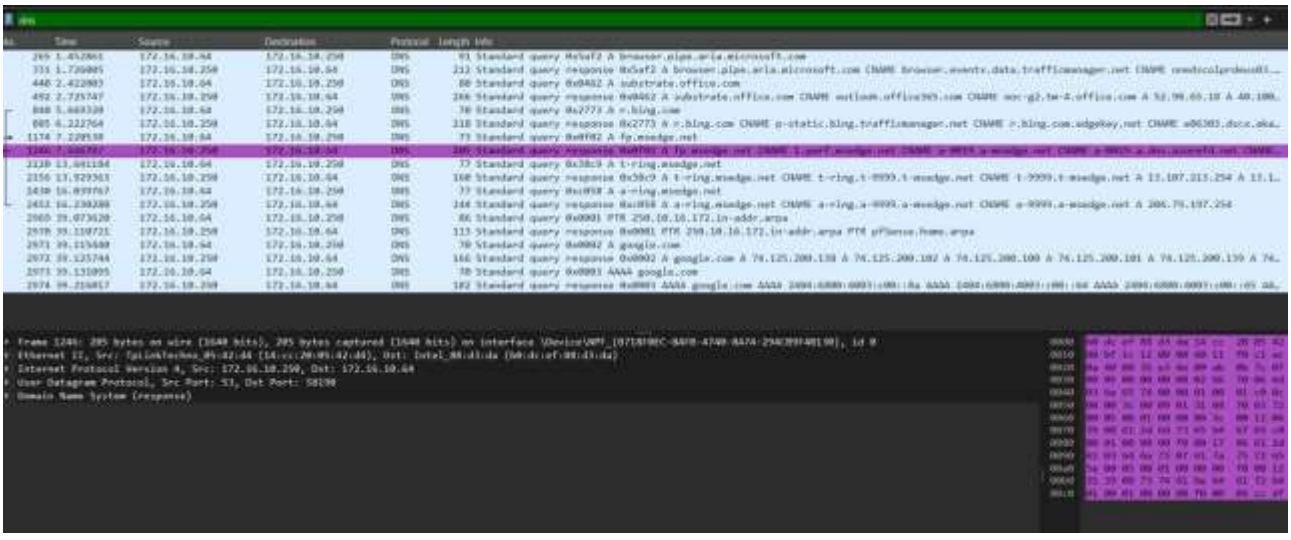
1. Open Wireshark to capture packets, open the web browser and use an application that uses UDP (perhaps DNS). Stop the packet capture.

2. Use the filter to display UDP packets (it is UDP segments).



3. Now you are ready to analyze UDP. The following questions can guide you to conduct your analysis. Make sure that you take enough screenshots and notes. You will use those to prepare your task submissions.

a. Pick one UDP packet from the filtered packets. Examine the number of fields that are in the header of the selected UDP packet. You can take a screenshot of the packet and name and explain the fields in the UDP packet.



- The UDP packet has a simple header with four main parts.

1. Source Port (16 bits)
- Value in Frame 1246: 53
- Explanation: This shows the port number of the program or service that sent the packet. Here, port 53 is used by DNS (Domain Name System), so this packet is a DNS response.

2. Destination Port (16 bits)
- Value in Frame 1246: 58190
- Explanation: This shows the port number where the packet should be sent. Port 58190 is likely a temporary port that the client is using to get the DNS response.

3. Length (16 bits)
- Explanation: This shows the total size of the UDP packet, including both the header and the data. While the exact number isn't given, we know the packet is 205 bytes, so this includes the header (8 bytes) and the data.

4. Checksum (16 bits)
- Explanation: The checksum is used to check for errors in the packet. It helps ensure that the data hasn't changed during transmission. The exact value isn't provided, but it's calculated using the header, data, and some parts of the IP header.

Data (Variable Length)
- Explanation: This is the actual information in the packet. In this case, it's a DNS response, which includes the answer to a DNS query. The data size can be found by subtracting the 8-byte header from the total length shown in the length field.

b. Can you identify the length of each UDP header field from this UDP packet? You can indicate the length in bytes. You may want to examine the displayed information in Wireshark's packet content field.

- Each field in the UDP header has a specific length, which is consistent across all UDP packets:
  1. Source Port: 2 bytes (16 bits)
  2. Destination Port: 2 bytes (16 bits)
  3. Length: 2 bytes (16 bits)
  4. Checksum: 2 bytes (16 bits) The total size of the UDP header is 8 bytes.

c. Can you explain the value in the Length field? What exactly this value indicates? You can verify your answer by examining captured UDP packets.

- The Length field in a UDP packet indicates the total size of the packet, including the UDP header and the data payload. The value in the Length field is given in bytes. The Length field value is 205 bytes, it means that the entire UDP packet, including the 8-byte header and the data, is 205 bytes long. The payload data would then be 205 - 8 = 197 bytes.

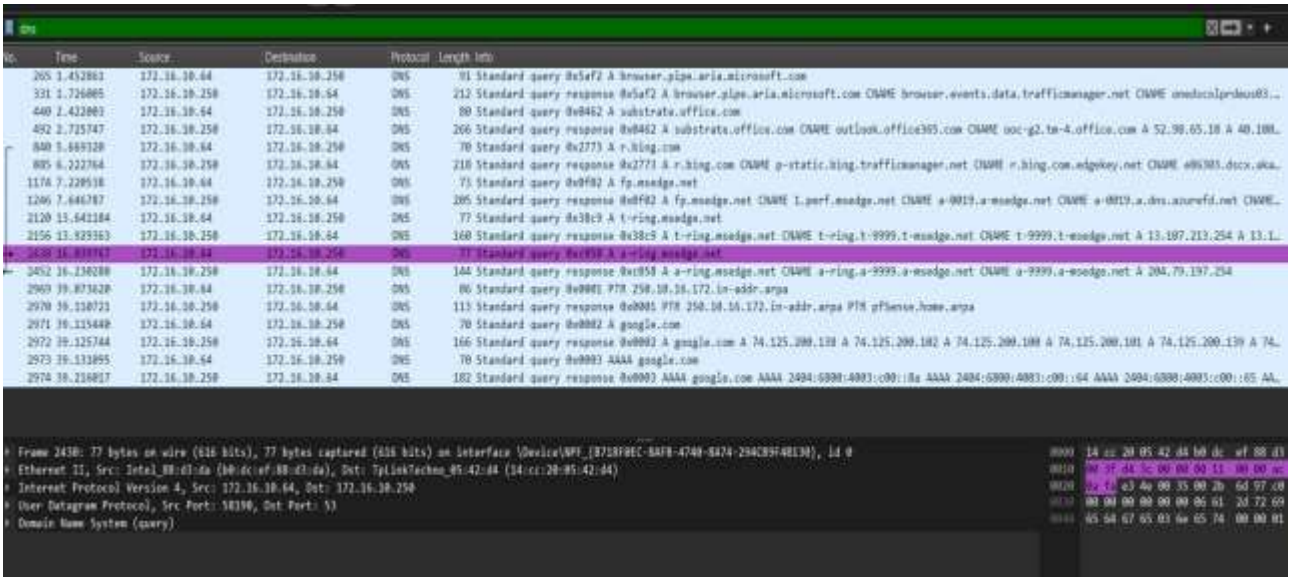
d. Is there a maximum number of bytes that we can include in UDP payload?

- The maximum size of a UDP packet is 65,535 bytes, which includes the header and the payload. Given the header is 8 bytes, the maximum size for the payload is 65,535 - 8 = 65,527 bytes. However, the actual maximum payload size can be limited by the underlying network protocols and infrastructure, such as the Maximum Transmission Unit (MTU) of the network.

e. Can you identify the protocol number given for UDP?

- The protocol number assigned to UDP in the IP protocol field is 17. This number identifies the UDP protocol at the IP layer.

f. Examine two consecutive UDP packets your host sends/receives. Explain the relationship between the port numbers in these two packets.



- In the two UDP packets provided:

1. Frame 2430:
- Source Port: 58190
- Destination Port: 53 - Type: DNS Query

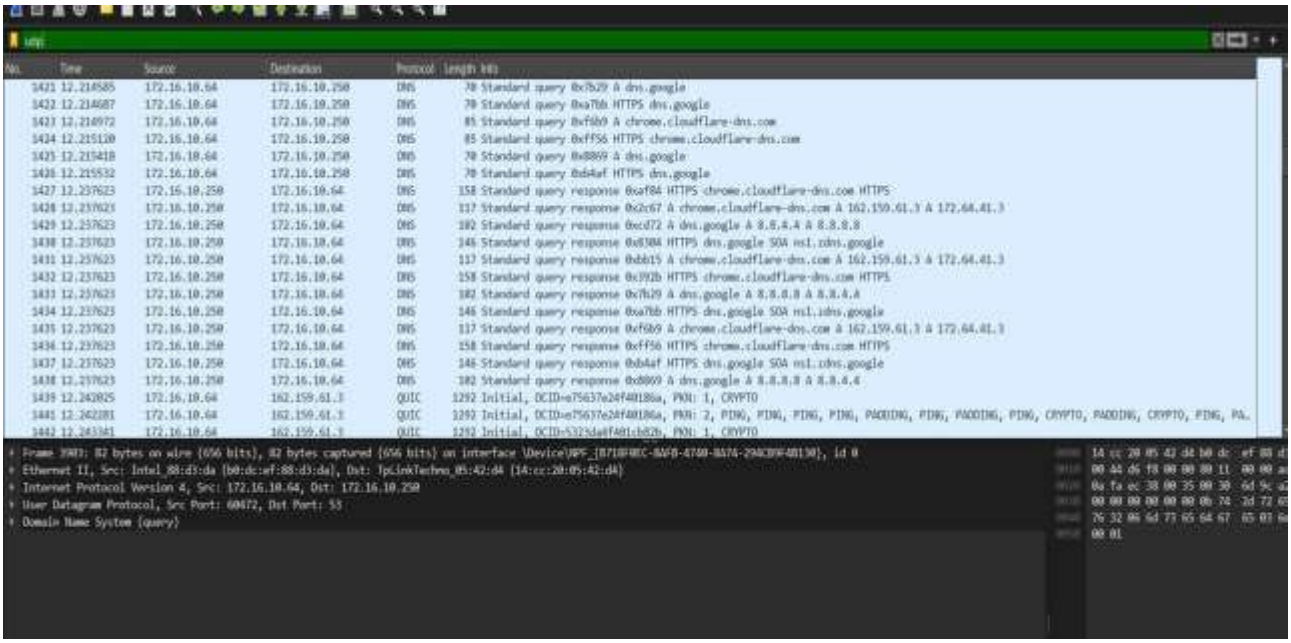
2. Frame 2452:
- Source Port: 53
- Destination Port: 58190 - Type: DNS Response

Relationship Between Port Numbers

- Source and Destination Ports: In the first packet (Frame 2430), the client (at IP 172.16.10.64) sends a DNS query from a high-numbered source port (58190) to the DNS server's standard port (53). In the second packet (Frame 2452), the DNS server (at IP 172.16.10.250) sends a response back to the client's source port (58190).

- Port Reversal: The roles of the source and destination ports are reversed between the query and the response. The client used port 58190 to send the query, and the DNS server responds to that port, while using its standard port 53 as the source.

g. Clear the cache and run another application such as MSTeams while using Web browsing. Capture packets of these two applications. Now you can analyze the captured UDP packets again. Do all UDP packets captured in Wireshark use the same port number? Explain your answer.



No, not all UDP packets in Wireshark will use the same port number. Because,

1. Different Apps Use Different Ports: Apps like MSTeams and web browsing use different ports for their UDP communication. For example:
- MSTeams uses various ports for audio, video, and data.
- Web browsing generally uses TCP but might use UDP for things like DNS or streaming, and those will have different ports.

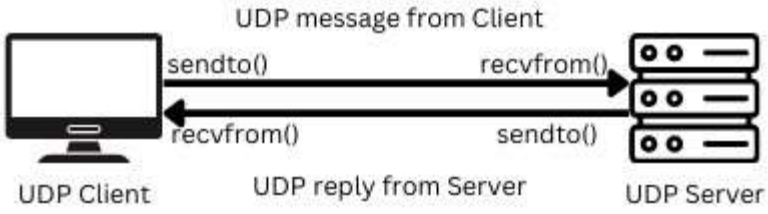
2. Port Numbers Change: UDP packets have source and destination port numbers that help identify the specific app or service. So, packets from MSTeams and web browsing will show different port numbers.

3. Multiple Ports per App: An app might use different ports for different functions. For instance, MSTeams might use different ports for video calls versus chat messages.

So, different port numbers are there for UDP packets from different applications.

Activity 3

1. First you need to draw a diagram to explain the operation of the client- server program that uses UDP.



UDP Client – Server diagram

2. Then one of you can develop the client side and the other can develop the server side of the program (you can also develop both end systems and demonstrate the outcome if you chose to do so)

```
from socket import *
serverPort = 11500
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', serverPort))
print("The server is listening")
while True:
    message, clientaddress = serverSocket.recvfrom(2048)
    ClientSendMessage1 = str(message) + " is received"
    serverSocket.sendto(ClientSendMessage1.encode(), clientaddress)
    print(ClientSendMessage1)
```

server.py code

```
from socket import *
serverName = "127.0.0.1"
serverPort = 11500
clientSocket = socket(AF_INET, SOCK_DGRAM)
message = "Hello SIT202"
clientSocket.sendto(message.encode(),(serverName, serverPort))
serverReply, serverAddress = clientSocket.recvfrom(2048)
print(serverReply.decode())
clientSocket.close()
```

client.py code

3. You need to show the output of the client- server program to demonstrate that your program works as expected. Please make sure to include the server and client codes and output (screenshots) in your Module 3 lesson review.

```python
from socket import *
serverPort = 11500
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', serverPort))

print("The server is listening")

while True:
    message, clientaddress = serverSocket.recvfrom(2048)
    message = message.decode()
    ClientSendMessage1 = str(message) + " is received"

    num_characters = len(message)
    print(f"Received message: '{ClientSendMessage1}' with {num_characters} characters")

    serverSocket.sendto(ClientSendMessage1.encode(), clientaddress)
```

Server.py code

```python
from socket import *
serverName = "127.0.0.1"
serverPort = 11500
clientSocket = socket(AF_INET, SOCK_DGRAM)

message = "Hello SIT202"
clientSocket.sendto(message.encode(),(serverName, serverPort))

serverReply, serverAddress = clientSocket.recvfrom(2048)
print(f"Received from server: {serverReply.decode()}")

clientSocket.close()
```

Client.py code

```
PS C:\Users\Raaid\OneDriv
hon/Python312/python.exe
The server is listening
```

Executing server.py and output

```
PS C:\Users\Raaid\OneDrive\Desktop\Raaid\Cyber Se
al/Programs/Python/Python312/python.exe "c:/Users
hon test/client.py"
Received from server: Hello SIT202 is received
```

Executing client.py and output

```
The server is listening
Received message: 'Hello SIT202 is received' with 12 characters
```

Updated output of server.py mentioning client message and character length

Above and Beyond Tasks

Modified the Python program you wrote to add the following functions:

1. The client sends a message “Hello” to the server
2. Once the server received the full message, sever responds with “Hello, What’s your name?”
3. When the client receives the full message from the server, Client should be able to enter the name via the terminal and send the name to the server.
4. When the server received the full message, server responds with “Hello name, Welcome to SIT202”. “name” should be the one that is received form the client.
5. All these received messages should be displayed in the relevant terminal. Please make sure to include the server and client codes and output (screenshots) in your task submission.

```python
from socket import *
serverPort = 11500
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', serverPort))

print("The server is listening")

while True:

    message, clientAddress = serverSocket.recvfrom(2048)
    message = message.decode()
    print(f"Received message: '{message}'")

    response = "Hello, What's your name?"
    serverSocket.sendto(response.encode(), clientAddress)

    name, clientAddress = serverSocket.recvfrom(2048)
    name = name.decode()
    print(f"Received name: '{name}'")

    finalResponse = f"Hello {name}, Welcome to SIT202"
    serverSocket.sendto(finalResponse.encode(), clientAddress)
```

Server.py code

```python
from socket import *

serverName = "127.0.0.1"
serverPort = 11500
clientSocket = socket(AF_INET, SOCK_DGRAM)

message = "Hello"
clientSocket.sendto(message.encode(), (serverName, serverPort))

serverReply, serverAddress = clientSocket.recvfrom(2048)
print(f"Received from server: {serverReply.decode()}")

name = input("Enter your name: ")
clientSocket.sendto(name.encode(), (serverName, serverPort))

serverReply, serverAddress = clientSocket.recvfrom(2048)
print(f"Received from server: {serverReply.decode()}")

clientSocket.close()
```

Client.py code

```
PS C:\Users\Raaid\OneDrive\Desktop\Raaid\Cyber Sec\CICRA\Degree\Year
& C:/Users/Raaid/AppData/Local/Programs/Python/Python312/python.exe
s/Week 5/Above and Beyond Client Server/server.py"
The server is listening
```

Executing server.py and output

```
Received from server: Hello, What's your name?
Enter your name: Raaid
Received from server: Hello Raaid, Welcome to SIT202
```

Executing client.py and output

```
The server is listening
Received message: 'Hello'
Received name: 'Raaid'
```

Updated output of server.py mentioning client message and name.

Computer Networks and Communication
Task 5.1P

Module Summary

1. Activity 1: Transport Layer Protocol

    - Conversation in Groups: The assignment was to talk about two packing scenarios that are comparable to TCP (Transmission Control Protocol) and UDP (User Datagram Protocol), two protocols used at the transport layer. Similar to UDP, which is used for quick but unreliable data delivery, the first scenario put speed before accuracy. Similar to TCP, which guarantees dependable and well-organized data transport, the second scenario concentrated on careful packing.
    - Comparison of TCP and UDP: According to the discussion, UDP is better suited for applications where speed is essential, like online gaming or live broadcasting, whereas TCP is best for situations demanding reliability, such big file transfers.

2. Activity 2: Packet Analysis using Wireshark

    - UDP Packet Analysis and capture: Using Wireshark, I had to record UDP packets and examine the header fields—Source Port, Destination Port, Length, and Checksum—of each packet. The research also included information on UDP protocol number, maximum payload size, and port number relationships between successive packets.
    - Recognizing Port Numbers that are Specific to Application: The test demonstrated that multiple UDP ports are used for different apps, including online browsing and Microsoft Teams. This demonstrates how UDP may handle a wide range of applications with flexibility.

3. Activity 3: Using UDP, a client-server application

- Department of Programs: The assignment was to create and present a UDP client-server application. The application demonstrated the useful application of the UDP protocol by requiring the client to communicate with the server by sending a message, waiting for a response, and then continuing the conversation.
- Improved Functionalities: Making the program capable of managing more interactive communication activities, like name exchanges and customized welcomes between the client and server, was one of the additional responsibilities.

Reflecting on the content

Upon reflection, I discovered that the module's content was both intellectually stimulating and practically applicable to comprehending the nuances of TCP and UDP in particular, two transport layer protocols. The trade-offs between network communication speed and dependability were clearly illustrated by drawing an analogy between packing scenarios and these protocols' features. The abstract ideas become more understandable and relatable as a result of this contrast. The practical experience using Wireshark was especially beneficial since it allowed academic information to be applied in a practical setting. My study of the structure and analysis of UDP packets has improved my understanding of data transmission and reception via networks.

Moreover, by enabling me to apply the ideas in a concrete manner, creating the client-server software with UDP strengthened the ideas. This hands-on approach helped me retain what I had learned and gave me valuable abilities that I will need in the future. In general, the assignments were made to close the knowledge gap between theory and practice, guaranteeing a comprehensive grasp of network protocols and their uses. The learning process was both successful and enriching because of the course team's clear emphasis on conceptual knowledge and practical application.

SIT202: Computer Networks and Communication
Leaning Evidence for Active Class Task 5

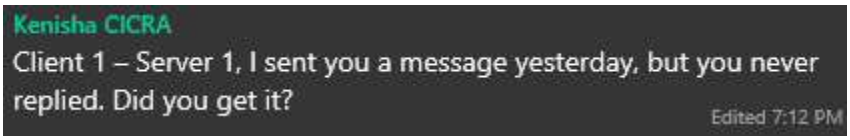
Name: Kenisha
Student ID: C23020001

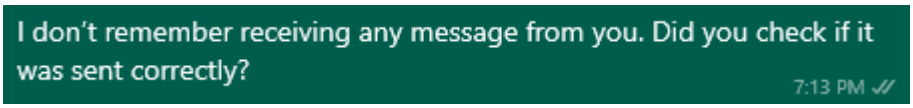
Members in this group activity task:

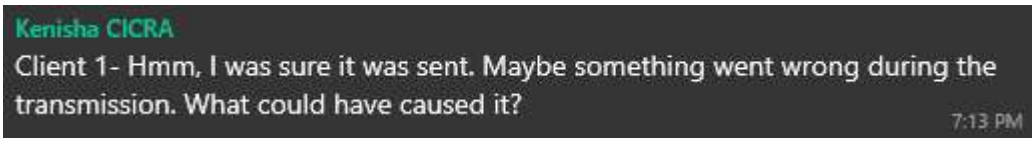
Nishad – C23110001
Kenisha – C23020001
Shekaina – C23110002
Raaid – C23020004
Pavithran – C22060015

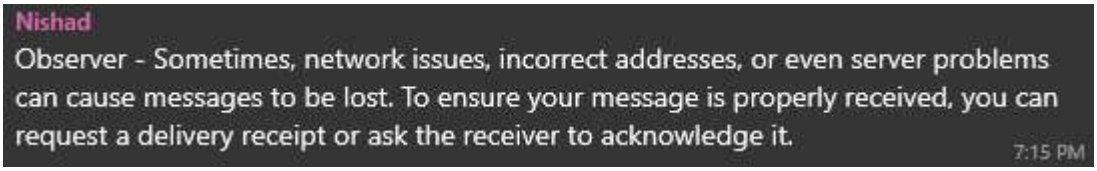
Activity 1

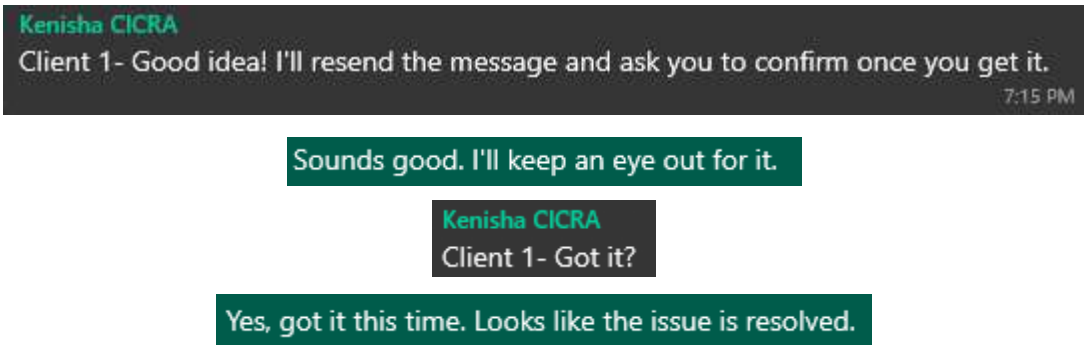
1. Assume one of your group members sent you a message. However, you have never received it. What went wrong? How do you make sure that the message is properly received?



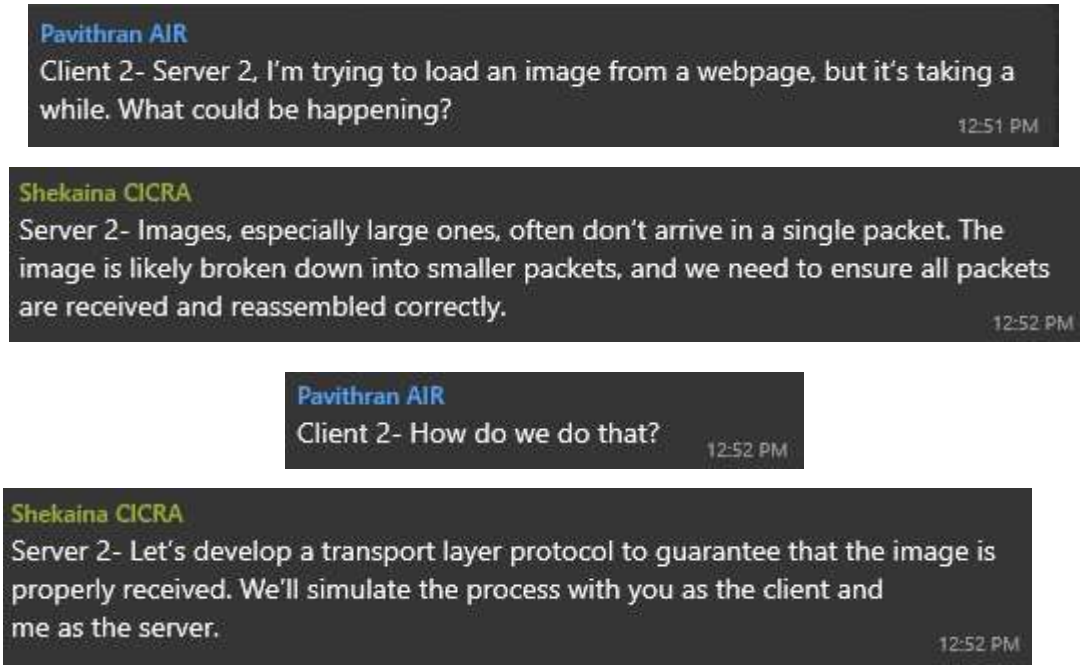




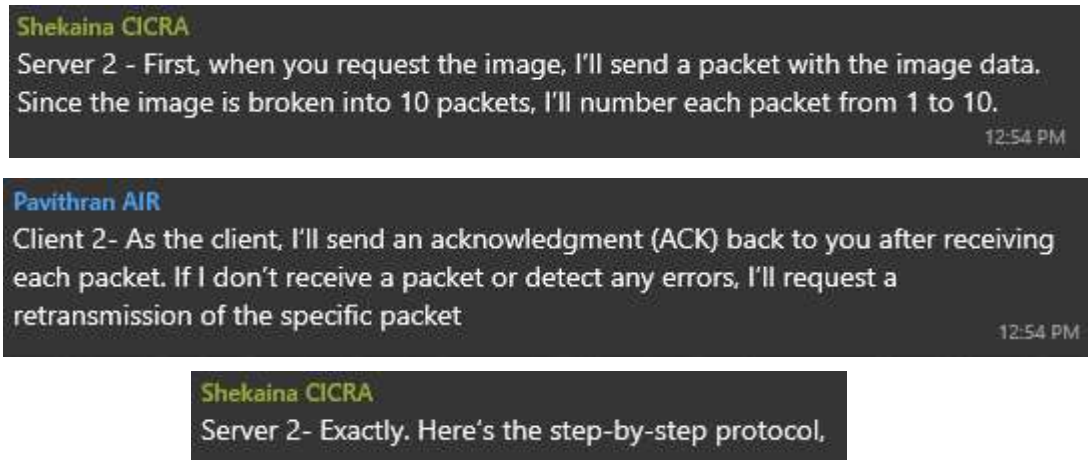




2. Assume you are trying to access a web page containing an image. Now, your job is to make sure that you get the image properly. The image cannot be received in one packet, and it will be broken down to 10 packets.



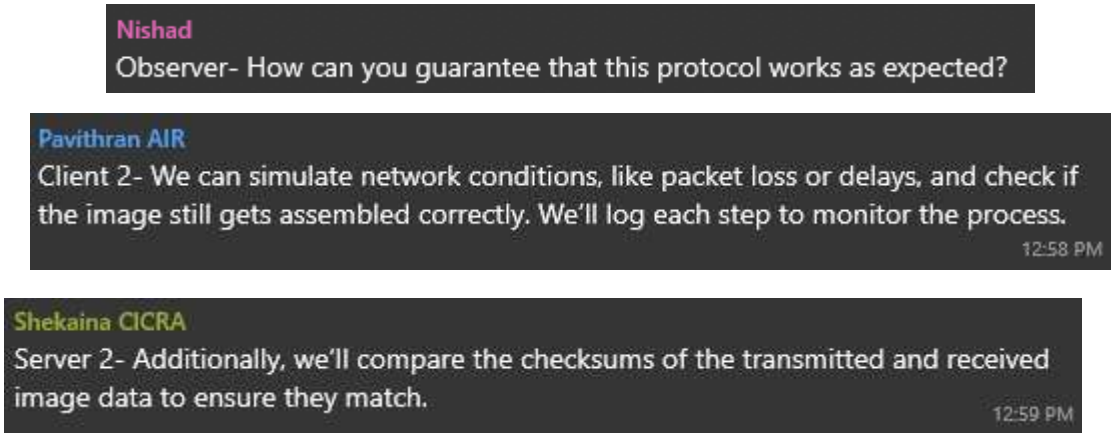
3. Partner up with one of your group members and develop a transport layer protocol to guarantee that the image is properly received by the client. One can act as the server where the webpage/image is stored and the other can act as the client who wants to receive the image (this means your group has 2 pairs of server-client networks).



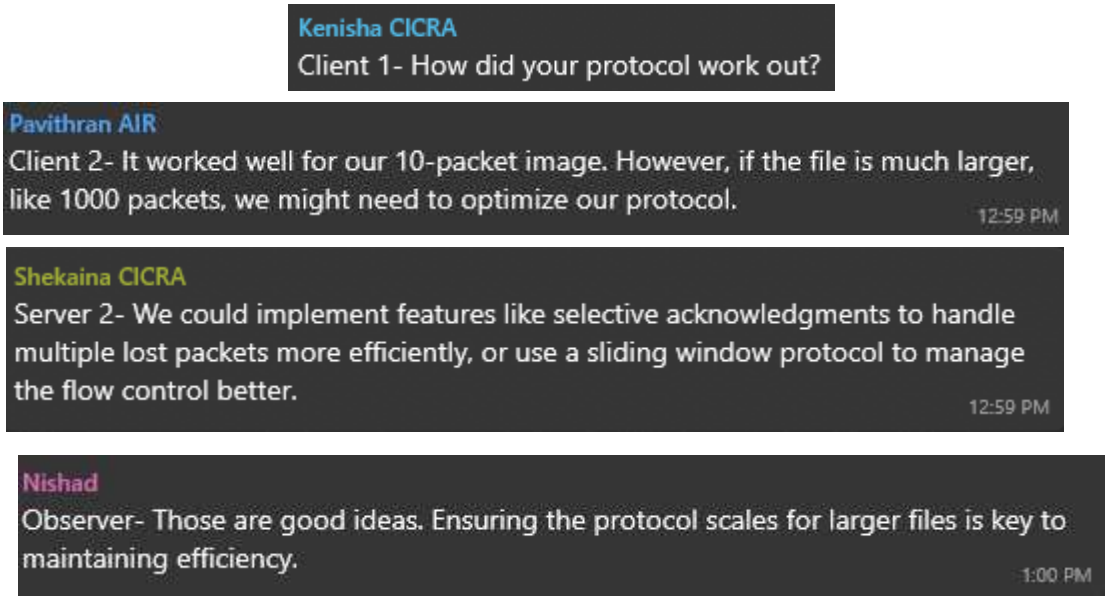
4. Write down protocol that you would use, including the communication between the client and server and all the messages that you are passing.

   - Client Request: Client 2 sends a request for the image.
   - Server Response: Server 2 begins sending packets, starting with packet 1.
   - Client ACK: Client 2 acknowledges the receipt of each packet (ACK 1, ACK 2, etc.).
   - Retransmission Request: If a packet is lost or corrupted, Client 2 requests retransmission of that specific packet.
   - Image Assembly: Once all packets are received, Client 2 assembles the image.

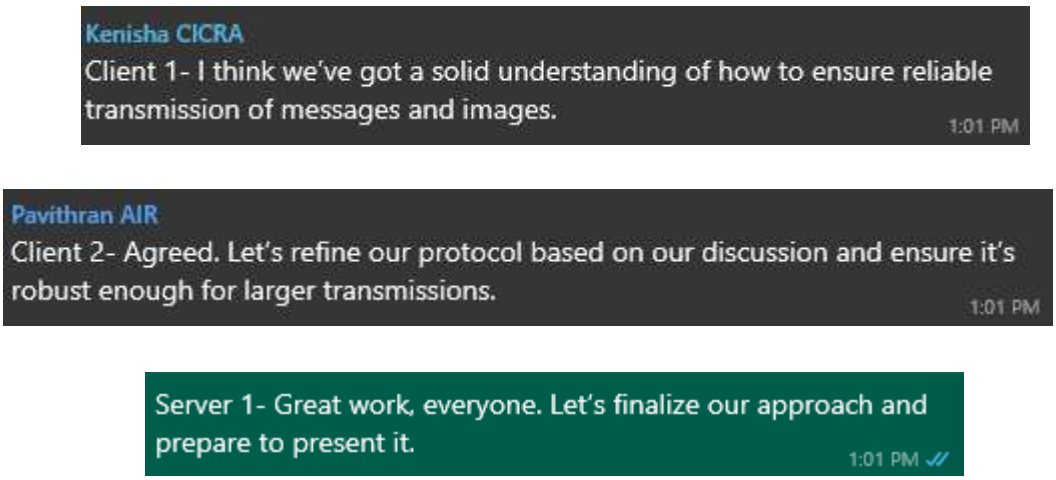
5. How do you guarantee that your protocol does the job as you expected?



6. Once you have finalized your protocol, check how other group members developed their protocols.

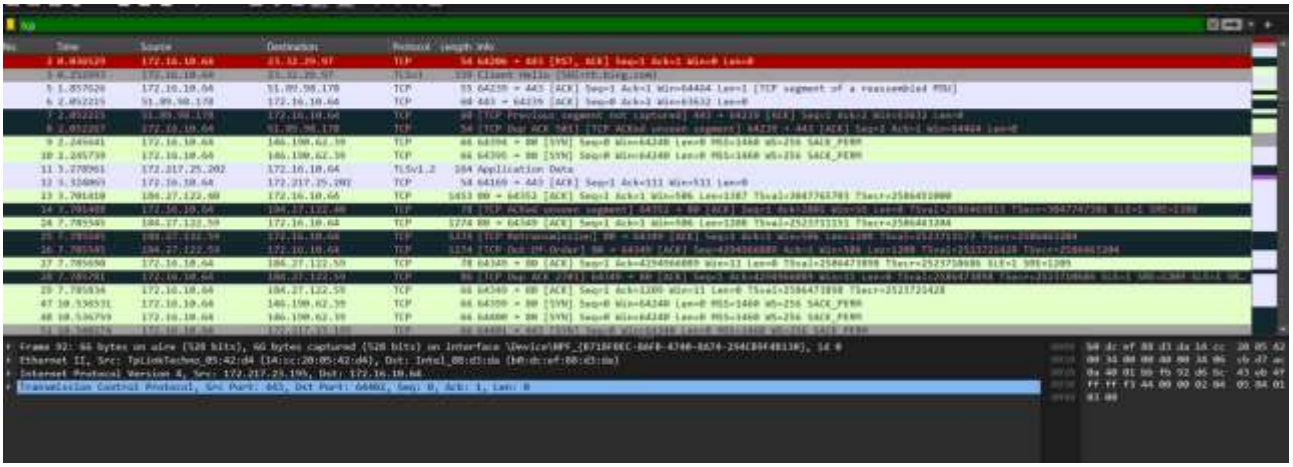


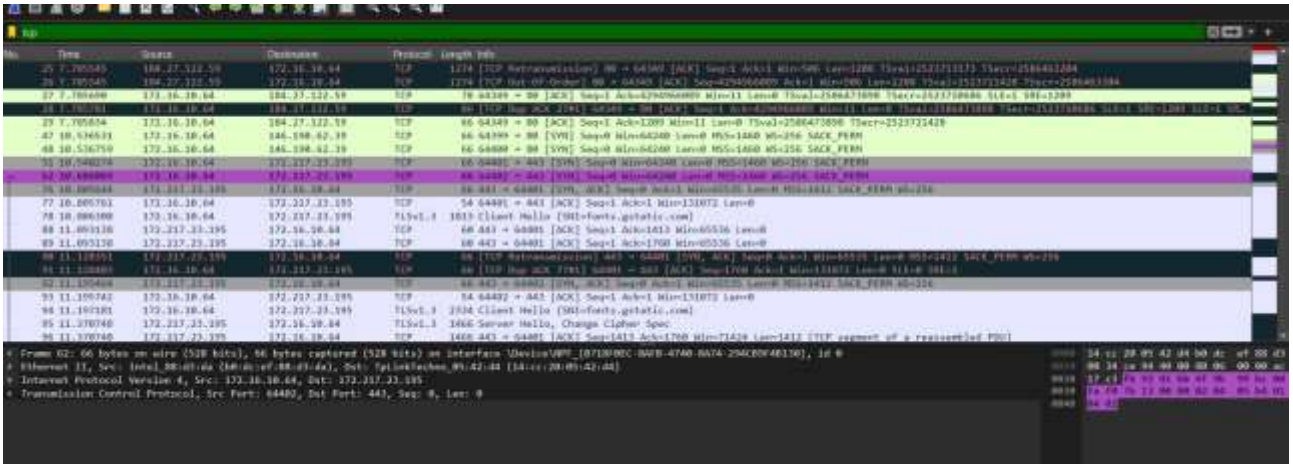
7. With your group, discuss whether your protocol valid when you need to access a large file that might be broken down into 1000 packets. If not, what are the changes you would like to make in your protocol?



Activity 2

1. You can analyze the TCP three-way handshake in Wireshark packet capture. Analyze the order of segments sent/received by two end systems to establish a TCP connection.
2. Analyze the segment headers used, their purpose and sizes.
3. Make sure you take the screenshots of TCP segments that will be used for your task submissions.

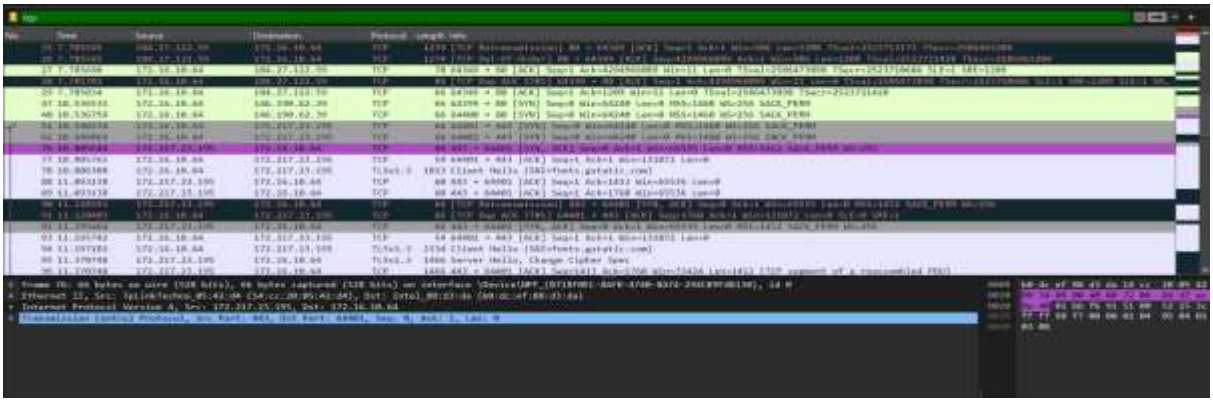




SYN Packet Analysis:

- **Source Port (Src Port)**: 64402
  - The client selected this port number in order to establish the connection. The port number is usually high and transient, used by the client.
- **Destination Port (Dst Port)**: 443
  - This shows that the server's port 443, which is frequently used for HTTPS traffic, is the destination of the SYN packet.
- **Sequence Number (Seq)**: 0
  - In this case, the client selected sequence number 0 as the starting point. The bytes of data sent from the client in this connection will be numbered starting at this point.

- **Length (Len)**: 0
    - Since this SYN packet is just used for the handshake and contains no data, the payload length is 0.



SYN-ACK Packet Analysis:

- **Source Port (Src Port)**: 443
    - This shows that the server is sending the SYN-ACK packet via port 443, which is frequently used for HTTPS.
- **Destination Port (Dst Port)**: 64401
    - The client-side port that started the connection is this one.
- **Sequence Number (Seq)**: 0
    - The server selected this as the first number in the sequence. This connection's sequence number begins at 0, which is typical.
- **Acknowledgment Number (Ack)**: 1
    - This is the number that verifies that the client sent the SYN packet and that it was received. The server sends the next expected sequence number, which is 0 + 1 = 1, in response to the client's likely initial sequence number of 0.
- **Length (Len)**: 0
    - Since the SYN-ACK packet is only being sent as part of the connection setup procedure and no actual data is being delivered yet, the payload length in this TCP segment is 0.

ACK Packet Analysis:

- **Source Port (Src Port)**: 64401
    - This is the client-side port number that was utilized to create the connection. It corresponds to the SYN-ACK packet's source port.

- **Destination Port (Dst Port)**: 443
    - This is the server's port number, which is 443 in this case, used for HTTPS.
- **Sequence Number (Seq)**: 1
    - Since 1 is the sequence number, it may be assumed that data transmission will use this as the subsequent sequence number. That being said, the sequence number in this instance hasn't increased over what was set in the SYN packet because the handshake is still in progress.
- **Acknowledgment Number (Ack)**: 1
    - The sequence number of the server from the SYN-ACK packet plus one represents the acknowledgment number that the client sent. A 1 for the acknowledgment number in the SYN-ACK packet signifies that the client is prepared to receive the following byte, which begins at sequence number 1, and validates receipt of the SYN-ACK packet, since the server's sequence number occurred to be 0.

.

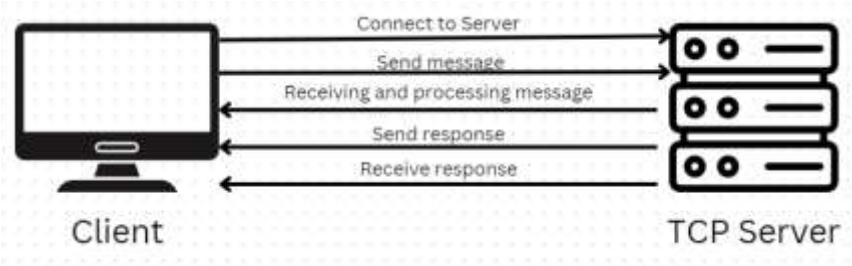
- **Length (Len)**: 0
    - The length of the payload in this packet is 0, consistent with the fact that this ACK packet is part of the handshake process and does not contain any data.

2. **Segment headers overview**

- **Source/Destination Port**: These numbers tell which programs or services are talking to each other.
- **Sequence Number**: This keeps track of where each piece of data is in the conversation, making sure everything arrives in the right order.
- **Acknowledgment Number**: This number confirms that data was received successfully.
- **Flags**: These tiny signals manage the connection, like starting it (SYN), confirming receipt of data (ACK), or ending it (FIN).
- **Window Size**: This shows how much data the receiver can handle at one time, helping control the flow of information.
- **Checksum**: This is like a safety check to make sure the data isn't messed up.
- **Options**: These are extra features that can improve how TCP works, like setting the biggest piece of data that can be sent at once (MSS).
- **Header Sizes**: The basic size is 20 bytes, but it can be 32 bytes or more if extra features are added.

Activity 3

1. First you need to draw a diagram to explain the operation and communication of the client-server program that uses TCP.



2. Then one of you can develop the client side and the other can develop the server side of the program (you can also develop both and demonstrate the outcome if you chose to do so)

```python
import socket

def start_server():
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    server_socket.bind(('localhost', 65534))

    server_socket.listen(5)
    print("Server is listening on port 65534...")

    client_socket, client_address = server_socket.accept()
    print(f"Connection from {client_address} has been established.")

    message = client_socket.recv(1024).decode()
    print(f"Received message: {message}")

    num_chars = len(message)
    print(f"Number of characters received: {num_chars}")

    uppercase_message = message.upper()

    response = f"{num_chars} {uppercase_message}"

    client_socket.send(response.encode())

    client_socket.close()
    print("Connection closed.")

if __name__ == "__main__":
    start_server()
```

server.py code

```python
import socket

def start_client():
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    client_socket.connect(('localhost', 65534))

    message = "Hello SIT202"
    client_socket.send(message.encode())
    print(f"Sent message: {message}")

    response = client_socket.recv(1024).decode()

    print(f"Received response: {response}")

    client_socket.close()

if __name__ == "__main__":
    start_client()
```

client.py code

3. You need to show the output of the client- server program to demonstrate that your program works as expected. Please make sure to include the server and client codes and output (screenshots) in your lesson review.

```
Server is listening on port 65534...
```

TCP server listening for any messages

```
Sent message: Hello SIT202
Received response: 12 HELLO SIT202
```

Message sent to server and received back from server

```
Server is listening on port 65534...
Connection from ('127.0.0.1', 55441) has been established.
Received message: Hello SIT202
Number of characters received: 12
Connection closed.
```

Complete output from TCP server

Above and Beyond Tasks

- Discuss the differences between the following three protocols and the reasons for using these protocols.
    - Stop-and-wait
    - The Stop-and-wait protocol is a simple Automatic method where the sender sends one packet and waits for acknowledgments from the receiver before sending the next frame.
    - Go-back-N
    - In this protocol, the sender can send multiple frames before needing an acknowledgment. Howver if there an error detected in the any of the frames in the window all frames in that window must be retransmitted
    - Selective repeat
    - Selective repeat can also send multiple frames before receiving an acknowledgment and if there's an error in a frame, only thst frame will be retransmitted, not the entire window. Selective repeat protocol is therefore considered an improvement from Go-Back-N protocol.

- Why do we need congestion control in TCP? Explain your answer.
    - Congestion control is needed in TCP for maintaining network stability and ensuring efficient us of available bandwifth. Withput congestion control, network instabilities can occur due to uncontrolled data flow where congestion collapses occur.

Computer Networks and Communication Activity 5.1P Part 2

Module Summary

Key networking fundamentals were thoroughly understood in this session, with a focus on the Transport Layer and the nuances of the Transmission Control Protocol (TCP). This lesson covered important topics such segment headers, the TCP three-way handshake, and a comparison of transport protocols like Selective Repeat, Go-Back-N, and Stop-and-Wait. Using Wireshark for packet analysis and creating unique transport layer protocols were only two of the hands-on activities that the course skillfully combined academic knowledge with. Ensuring network communication's efficiency, scalability, and dependability was highlighted by these operations.

In order to preserve network performance and stability, the curriculum also covered useful applications such client-server programming and congestion control techniques. In order to promote teamwork, problem-solving, and the practical application of principles in real-world circumstances, collaborative projects were essential to the learning process.

Reflecting on the content

When I think back on the material, the SIT202 module really helped me comprehend TCP and how important it is to dependable data transfer over networks. In particular, the mix of practical exercises and theoretical lectures really helped me understand the intricacies of networking. For example, using Wireshark to analyze TCP packets and creating custom protocols helped me understand how theoretical concepts may be applied in real-world scenarios. By exposing me to a variety of viewpoints and concepts, the collaborative components of the module—where we worked in groups to solve challenges and improve our approaches—greatly increased my learning experience.

Working together was essential to overcoming the difficulties associated with protocol creation and guaranteeing the effectiveness and scalability of our solutions. The curriculum gave me practical insights into the difficulties of network management and performance optimization because of its emphasis on real-world applications, especially through client-server programming and a grasp of congestion control. Overall, this program has given me a thorough and useful grasp of computer networking, enabling me to confidently tackle increasingly difficult subjects and practical difficulties in the industry.

SIT202– Computer Networks and Communication
Task 6.1P: Lesson Review

Module Summary

This curriculum employed a number of hands-on exercises and analytical exercises to provide a thorough understanding of computer networking and communication. The first part of the program covered the details of IP datagrams and their headers. Students were asked to identify important components such the IP version (IPv4 or IPv6), the transport layer protocol (TCP or UDP), and the source and destination IP addresses. The investigation also included the payload size, the IP header size, and other important elements like the Explicit Congestion Notification (ECN) and Time to Live (TTL). The assignments made it easier to see how important each IP header field is and how they all work together to provide effective data transfer across networks.

The lesson also looked at how networking ideas might be used in real-world scenarios through interactive exercises with Cisco Packet Tracer. The assignment given to the students involved setting up a simulated network environment with several Local Area Networks (LANs) connected by a router. Every group member set up their devices to establish connection between various LANs, assuming a role (such as PCs and routers). Through this exercise, participants could learn how to configure gateways, IP addresses, and other network settings that are essential for successful data transfer. In order to evaluate the network configuration and comprehend the communication flow between devices, students also tested connectivity using the "ping" command.

An analogy exercise comparing the procedures of delivering data via a network to those of sending a parcel through the postal service was also included in the module. This contributed to a better understanding of how data packets travel through multiple sorting centers and are eventually reassembled to reach their destination, much like a parcel travelling to a recipient in a foreign nation. The exercise demonstrated how network protocols control data flow and underlined the intricacy and cooperation needed for both procedures.

In order to ensure connectivity through further configuration and testing, advanced activities urged students to add more devices to their network, such as a new PC. Through this activity, students were able to investigate more complicated ideas that are essential for larger and more intricate network settings, like subnetting, routing, and device management inside a network.

Reflecting on the content

Upon reflection, the mix of academic knowledge and practical application yielded the most important learning from this subject. I developed a greater grasp of how data is managed and sent over various networks by taking part in activities that entailed configuring networks and analyzing IP datagrams. Through the exercises, participants gained understanding of the significance of each procedure and its function in guaranteeing dependable, safe, and effective communication. The methodical process of creating a network in Cisco Packet Tracer had a special effect since it necessitated the application of networking concepts in a realistically replicated setting.

This lesson also assisted me in connecting new information to previously understood ideas. For example, prior knowledge of how data is processed at different tiers of the OSI model was necessary to grasp the relationship between the IP header fields and their function in data transmission. The practical exercises improved my ability to solve problems and reaffirmed the significance of paying close attention to details when configuring networks. Furthermore, the data transmission complexity were clarified by using the analogy of shipping a parcel, which also made it simpler to understand how packets go via different network devices and protocols in order to reach their destination.

The course team's overall goal appeared to be to present a comprehensive understanding of networking by fusing theory and real-world application. This method aids in preparing students for networking issues that arise in the real world, where both practical knowledge and in-depth understanding are crucial. Anyone interested in a career in network administration, cybersecurity, or any other job involving the management and upkeep of computer networks ought to understand these ideas. In addition to imparting necessary technical knowledge, the module helped me build critical thinking and problem-solving abilities that would be very useful in my future profession.

SIT202: Computer Networks and Communication

Leaning Evidence for Active Class Task 6

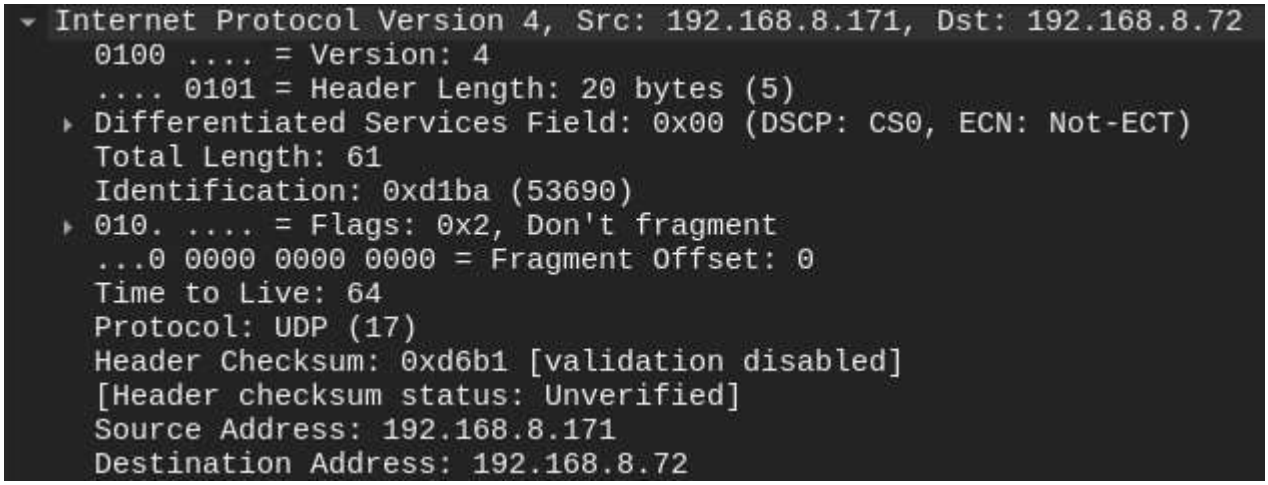
Name: Kenisha Corera
Student ID: C23020001

Members in this group activity task:
Nishad – C23110001
Kenisha – C23020001
Shekaina – C23110002
Raaid – C23020004
Pavithran – C22060015

Activity 1

Using Website http://httpforever.com/ and tracking DNS protocol for this activity,

1. Examine the IP datagram's header. Can you identify the IP address of your device and the IP address of the destination host?



- Source IP Address is 192.168.8.171 Destination IP Address is 192.168.8.72

2. Can you identify the version of IP addresses?

- The version is IPv4

3. By examine the IP header, can you identify the transport layer protocol used? Does this match with the transport layer protocol listed in in the packet details window?

- The protocol used in the transport layer is UDP and it does match with the protocol listed in the packet details window.

4. What is the size of IP header in Bytes? What is the size of the payload (in Bytes) of this IP datagram? How did you calculate the size of the payload?

• IP header size is 20 bytes. The size of the payload is 41 bytes (61 – 20)

5. Can you check whether this IP datagram is fragmented or not? Explain your answer.

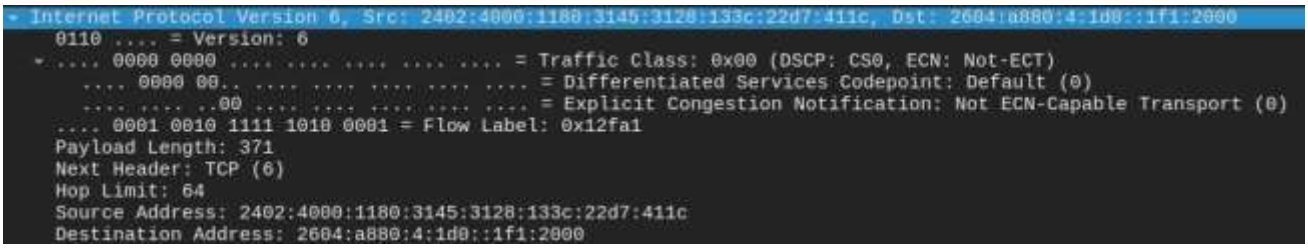
• This IP datagram is not fragmented because the flag value has been set to '010' where the 1 in the second bit means fragmentation is disallowed.

6. Examine other important fields in the IP header.

- Explicit Congestion Notification (ECN) which is a feature present in TCP/IP networks that marks packets that transit parts of the network experiencing high levels of congestion. ECN is set to '00' for this instance indicating that this datagram is not ECN capable.
- Time to Live is '64', which means that the datagram can traverse 64 more hops before being discarded.

7. Now, conduct a similar analysis using a different type of packet (if you have used HTTP before, now you can use DNS) and compare the findings.

Analyzing HTTP protocol now,



- Source IP Address is 2402:4000:1180:3145:3128:133c:22d7:411c
  Destination IP Address is 2604:8880:4:1d0::1f1:2000

- The version used in this datagram is IPv6

- The protocol used for the Transport Layer protocol is TCP

- The IP header size is 40 bytes and the payload size given is 371 bytes

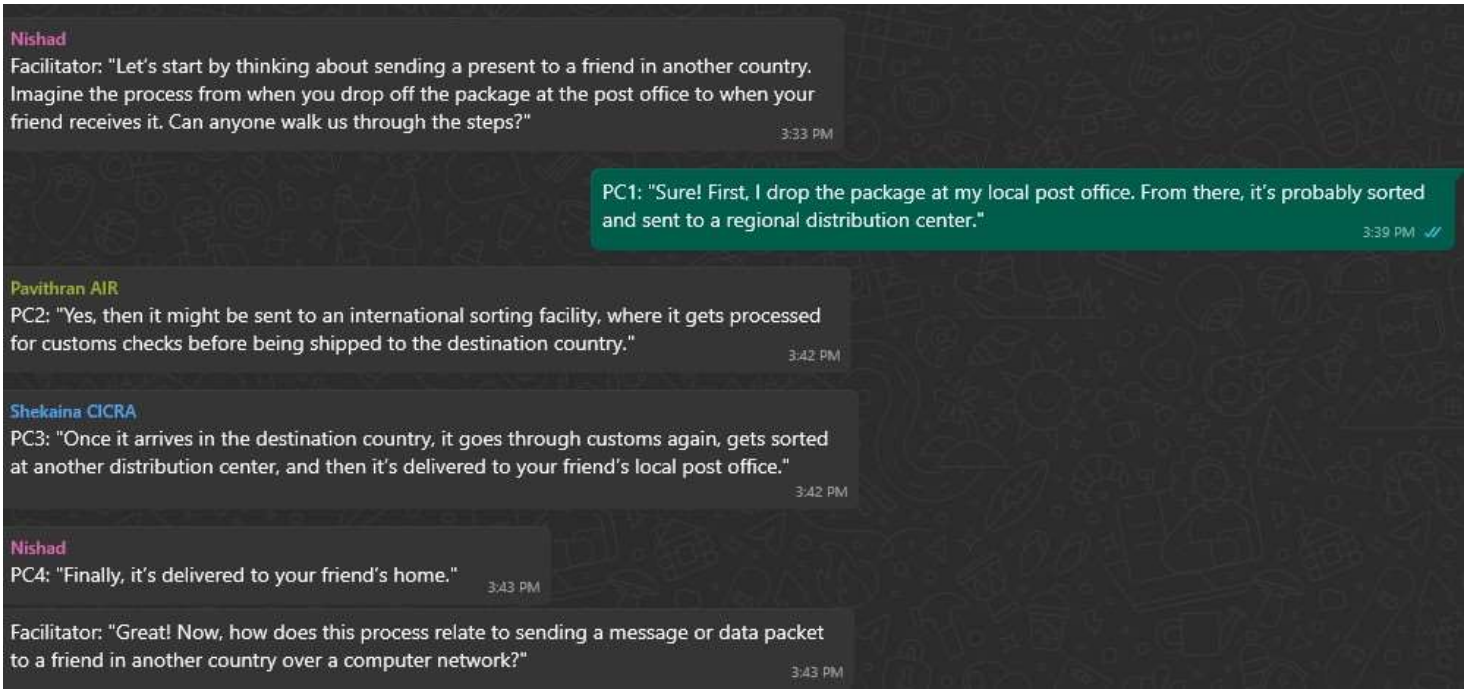
- This datagram is not fragmented since there's no indication of it or the presence of a fragmentation header.

- The hop limit is 64 (similar to Time to Live in IPv4)

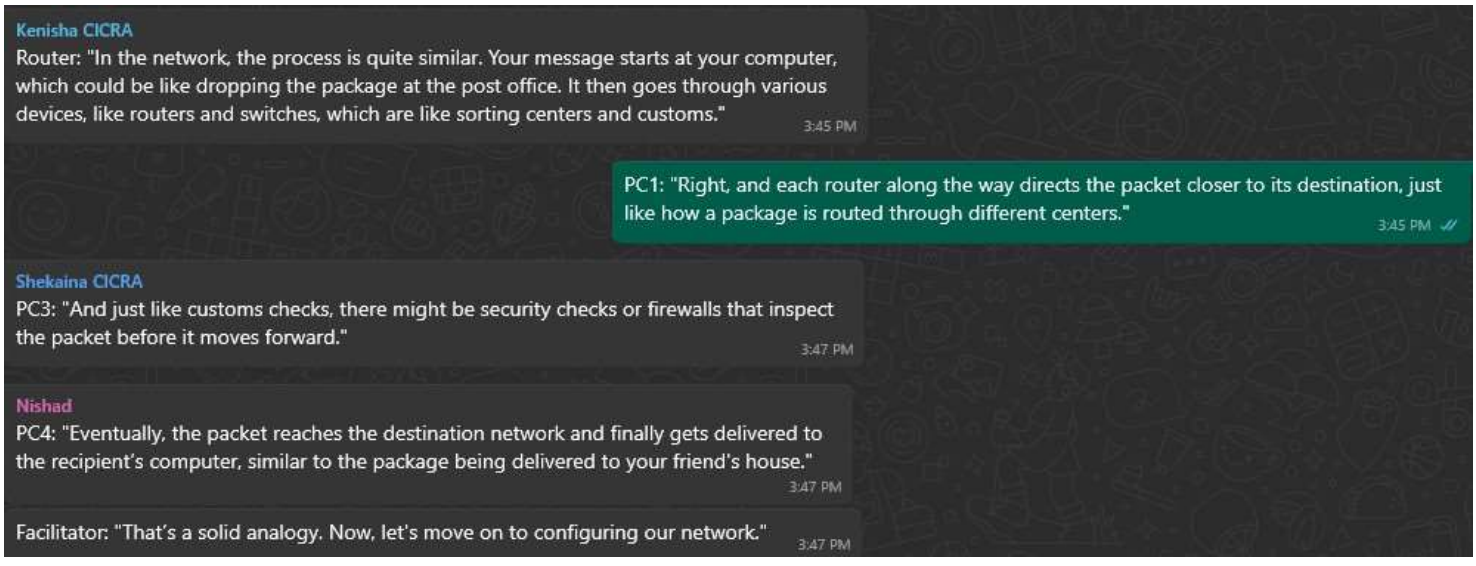
Activity 2

1. Assume you are sending a present to a friend in another country. Can you list the various places and steps that your parcel would go in the postal system before it reaches your friend?

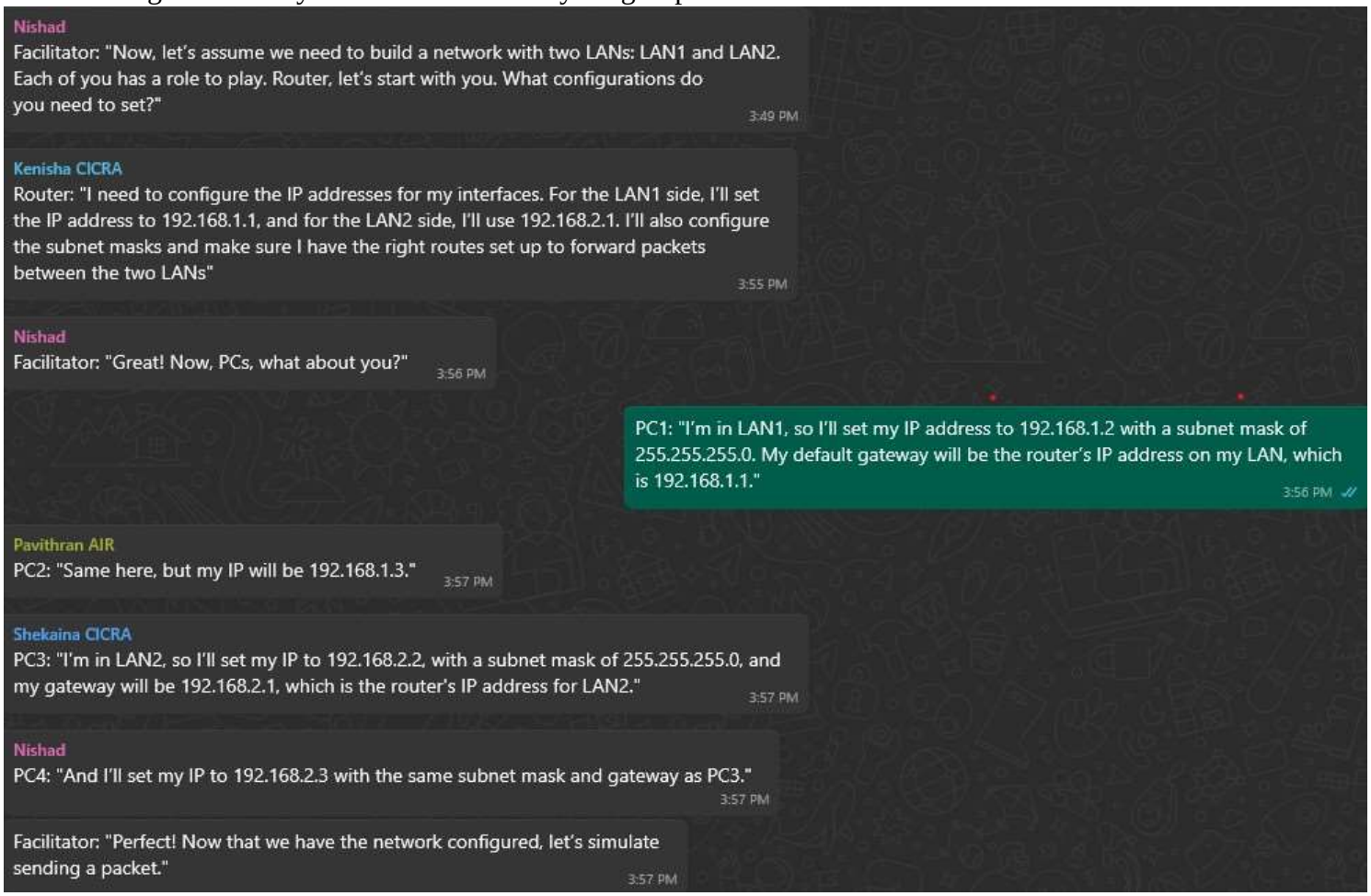
Router: Manages traffic between LAN1 and LAN2.
PC1 and PC2: Belong to LAN1.
PC3 and PC4: Belong to LAN2. 3:30 PM



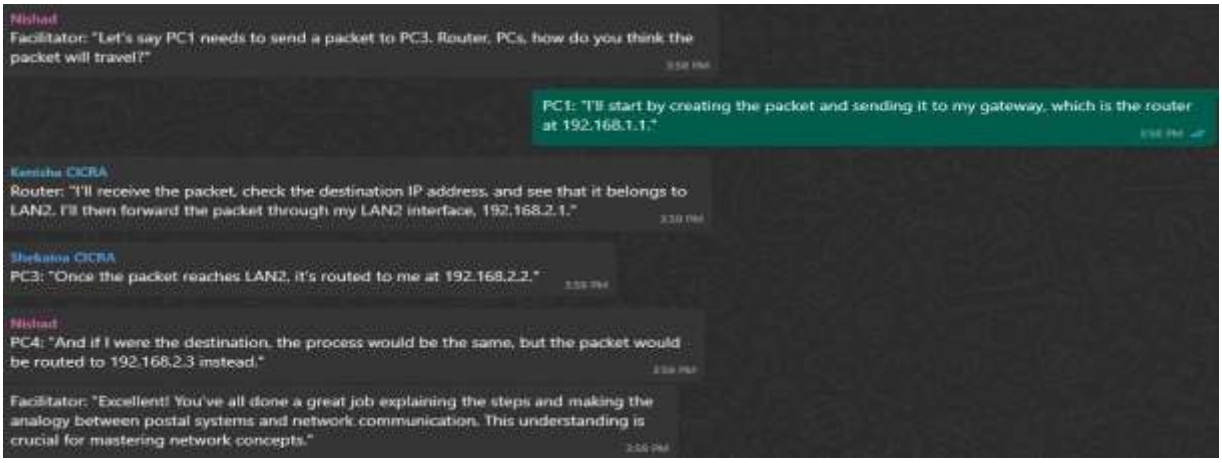
2. How this analogous to a situation where you want to send a message to a friend in another country over the computer networks?



3. Assume we need to build the following network with two LANs (LAN1 and LAN2). Each group member has a role to play. One group member can be the router and four other group members could be PCs (PC1 and PC2 belong to LAN1 and PC3 and PC4 belong to LAN2). Each device needs to set their own network configuration. The Router needs to set its interfaces/port and PCs need to set its IP address and gateways to be able to make a communication between two LANs. Discuss the configurations of your own device with your group members.



4. Assume PC1 needs to send a packet to PC3, discuss the steps that the packet needs to go through to reach to PC3.



Activity 3

1. Implement the above-mentioned network in Cisco Packet Tracer. You need to determine the IP addresses of all PCs depending on the LAN that they belong to (you have done this in Activity 2)
2. Once all the devices are configured and connected properly, verify the connectivity using command prompt "ping" in one of the PCs (ex: if PC3's IP address is 192.168.1.5 then from PC1's command prompt we can type "ping 192.168.1.5" to verify the connection) 3. Use the simulation mode to verify the steps that you have discussed in Activity 2 Step 4.
4. Make sure to take screenshots that you can use for task submissions.

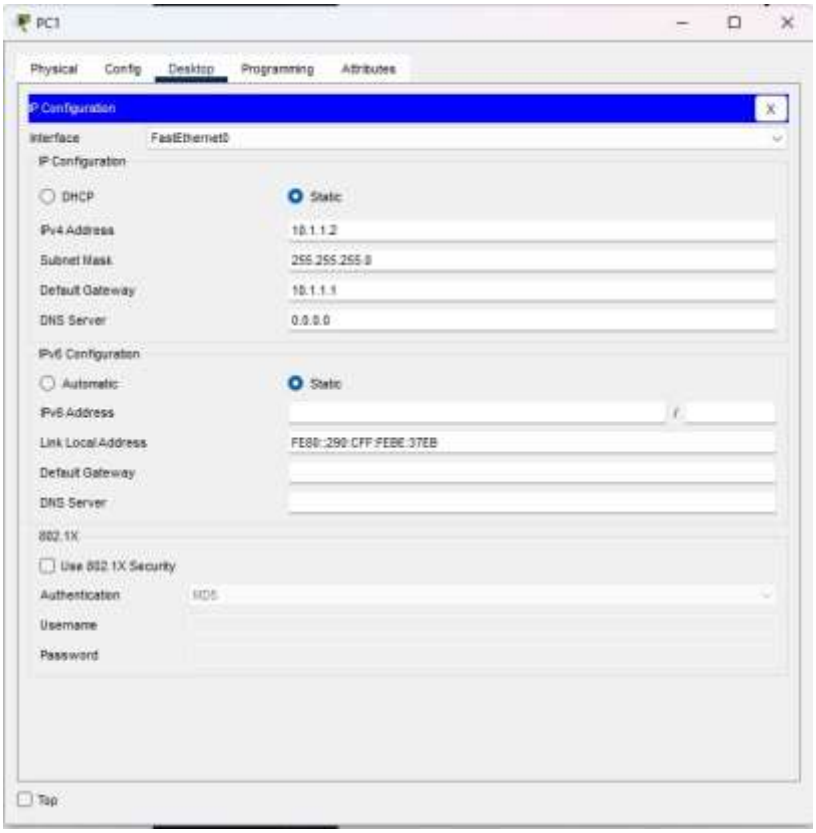
Above and Beyond Tasks

Connecting more devices to the network you built in Activity 3.

1. PC1 in the above diagram now wants to send a message to another PC (PC5) with the IP address of 198.168.2.4.
2. Discuss how PC5 is connected to the current network.
3. Add PC5 and other devices (if necessary) to the network you built in Activity 2 and verify the connectivity using "Ping" in one of the PCs.

Both Activity 3 and Above and Beyond Tasks were done together.

Configuration of IP addresses in Router



IP configuration of PC1

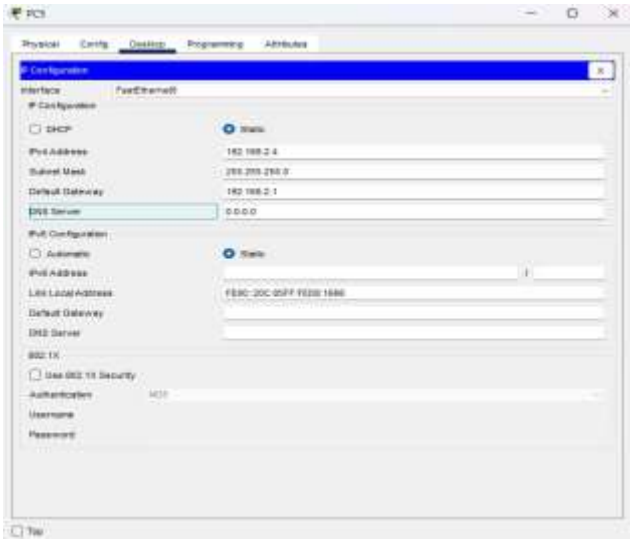
IP configuration of PC2

IP configuration of PC3

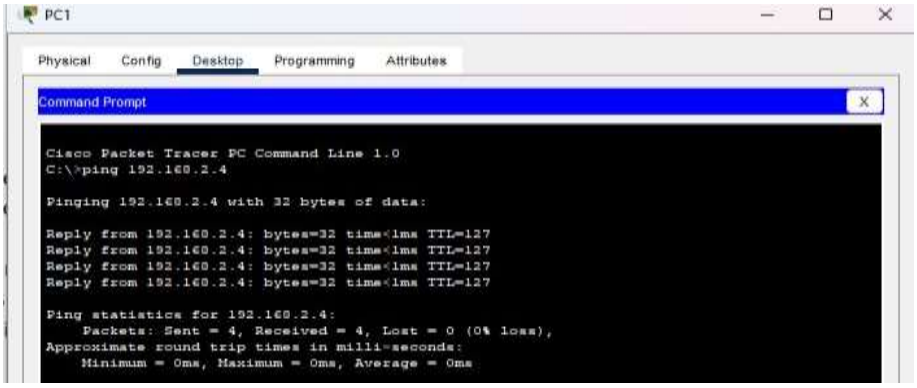
IP configuration of PC4

```
C:\>ping 192.168.1.5

Pinging 192.168.1.5 with 32 bytes of data:

Reply from 192.168.1.5: bytes=32 time<1ms TTL=127
Reply from 192.168.1.5: bytes=32 time<1ms TTL=127
Reply from 192.168.1.5: bytes=32 time<1ms TTL=127
Reply from 192.168.1.5: bytes=32 time<1ms TTL=127

Ping statistics for 192.168.1.5:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Ping from PC1(IP address: 10.1.1.2) to PC3 (IP address: 192.168.1.5)



IP configuration of PC5



Ping from PC1(IP address: 10.1.1.2) to PC5 (IP address: 192.168.2.4)

Network architecture

Router1 (2911) was connected to three switches (switch-PT) using copper straight through cables (gig ethernet) and each switch was connected to 2 PCs (excluding switch 3 which was connected to only 1 PC which was PC5) using copper straight through cables (fast ethernet).

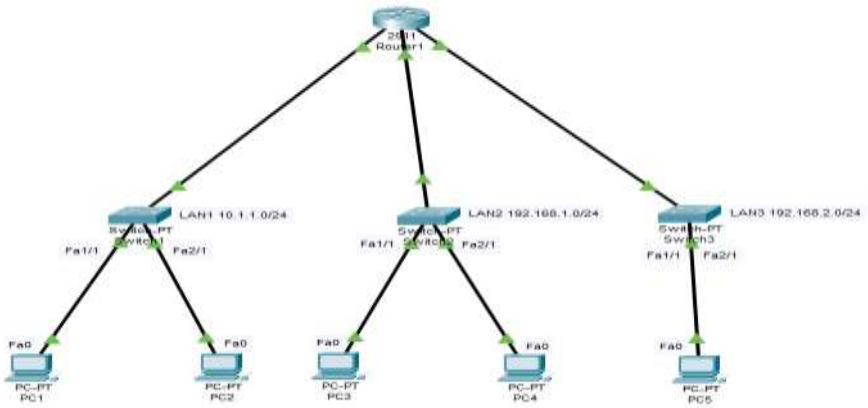
There are in total 3 LAN connections.

LAN1 – 10.1.1.0/24

LAN2 – 192.168.1.0/24

LAN3 – 192.168.2.0/24

PC5 (192.168.2.4/24) is connected under LAN3.



Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC1	PC5	ICMP		0.000	N	0	(edit)	(delete)
	Successful	PC5	PC3	ICMP		0.000	N	1	(edit)	(delete)
	Successful	PC2	PC5	ICMP		0.000	N	2	(edit)	(delete)

These are the successful attempts in verifying connectivity with PC5.

SIT202: Computer Networks and Communication
Leaning Evidence for Active Class Task 7

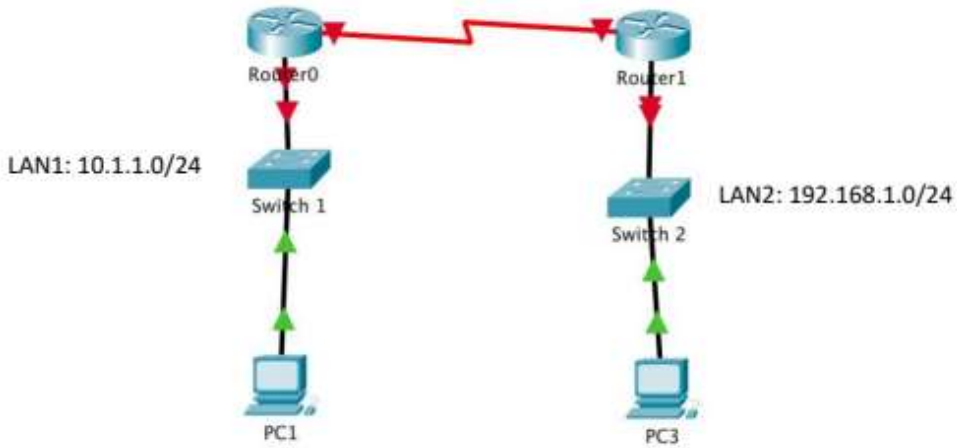
Name: Kenisha Corera
Student ID: C23020001

Members in this group activity task:

Nishad – C23110001
Kenisha – C23020001
Shekaina – C23110002
Raaid – C23020004
Pavithran – C22060015

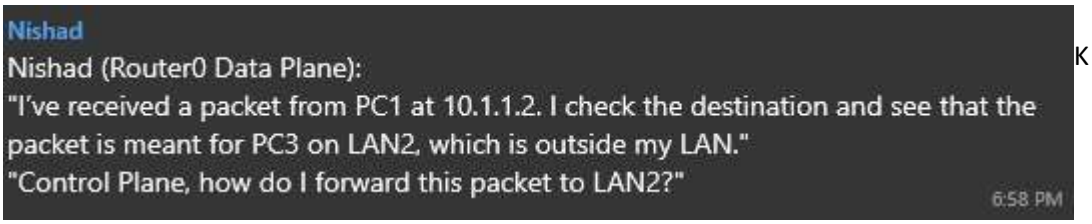
Activity 1

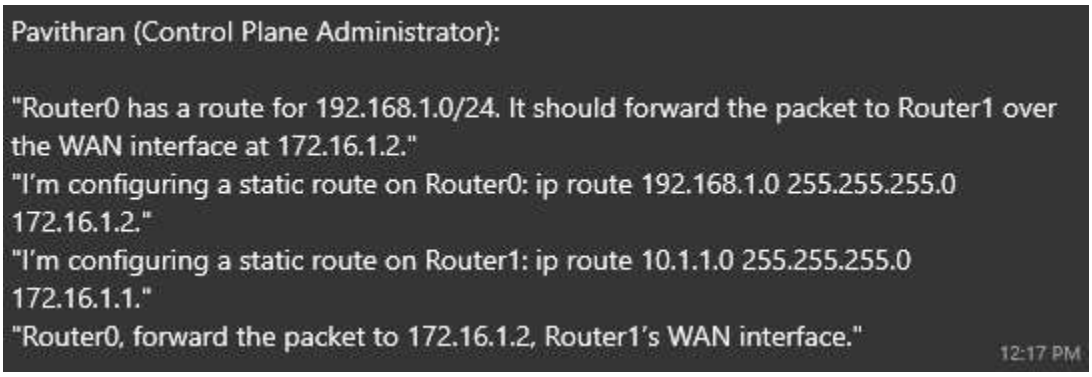
1. Assume your network has two routers and a separate LAN connects to each router as shown in the following figure.

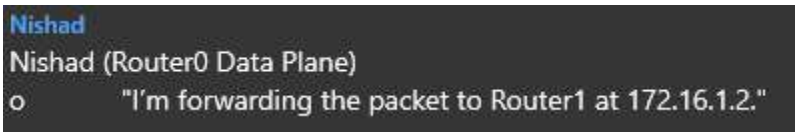


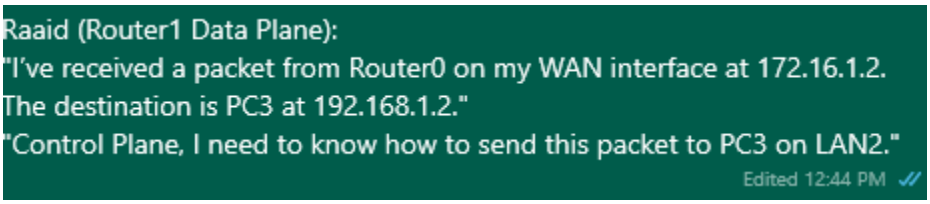
2. Now we are ready to do a role play. Two group members can act as LANs (one member for each LAN). They are responsible for all the PCs connected to their corresponding LAN. Two group members (or one member) are acting as data plane of Router 0 and Router 1. The remaining member is acting as the control plane of both routers.
3. Now assume PC1 wants to send a message to PC3, you need to work together to support the request. Each device needs to list your configurations.

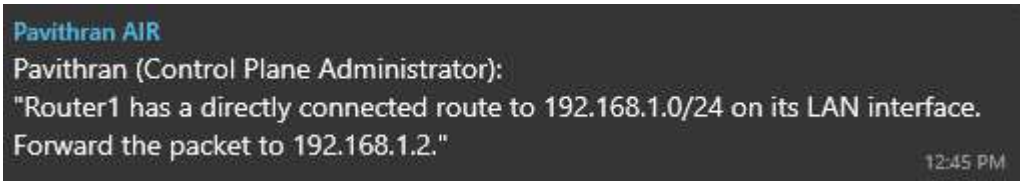
1. Kenisha (LAN1 Administrator - PC1):
"I'm configuring PC1 with the following details: IP address 10.1.1.2, subnet mask 255.255.255.0, and default gateway 10.1.1.1."
"PC1 is ready to send a packet to PC3, with the destination address 192.168.1.2."
"I'm sending a packet to the default gateway (10.1.1.1), addressed to PC3 at 192.168.1.2."
5:34 PM

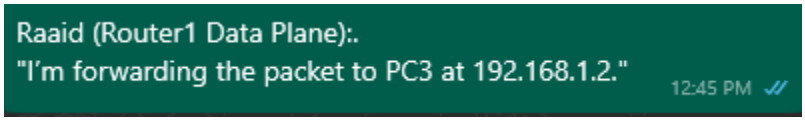












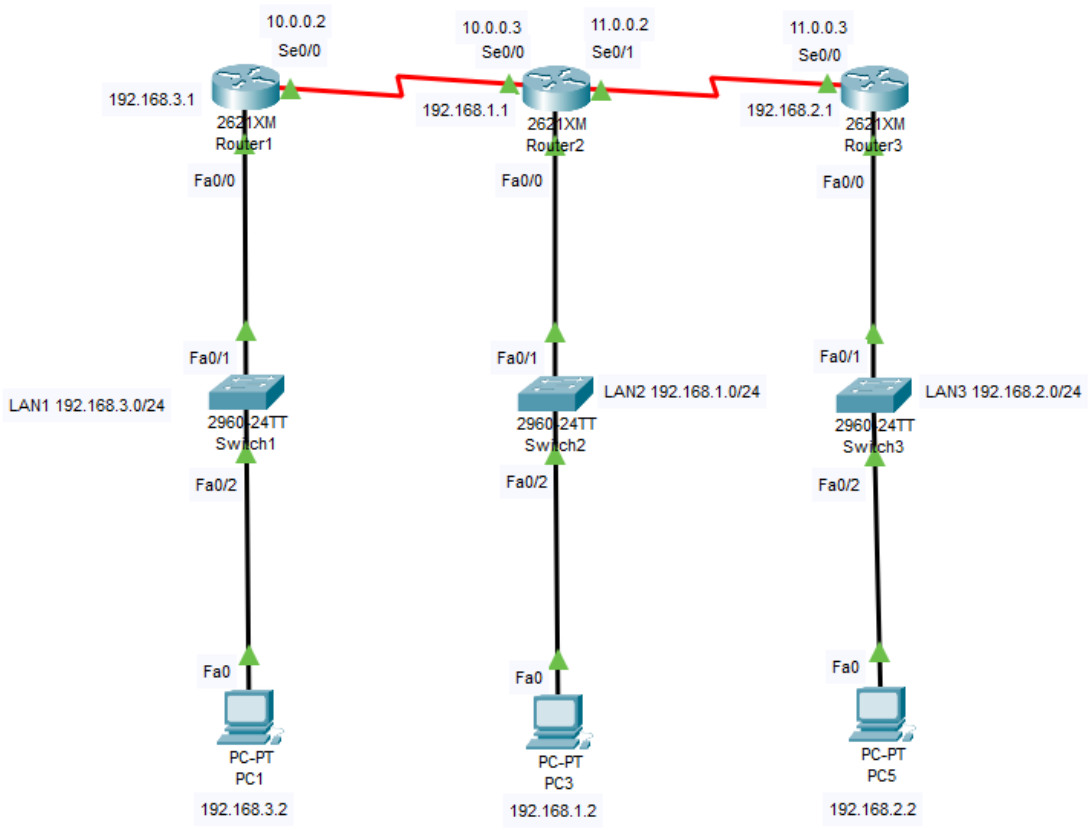
4. Discuss
    a. How many subnets are there?
        - There are three subnets,
            - LAN1 – 10.1.1.0/24 (PC1)
            - LAN2 – 192.168.1.0/24 (PC2)
            - WAN – 176.16.1.0/30 (subnet for routers)

b. The responsibilities of the control plan of the routers. Tips: How router 0 knows about the LAN 2 and router 1
   - Managing the routing table
   - Determining path
   - Static route configuration

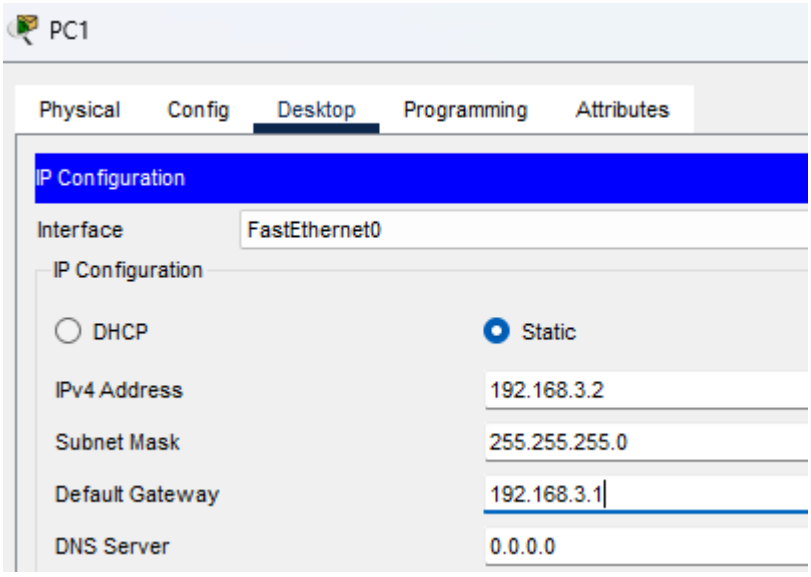
5. You are going to use these discussions for Activity 2 and 3.
   - Completed

Activity 2

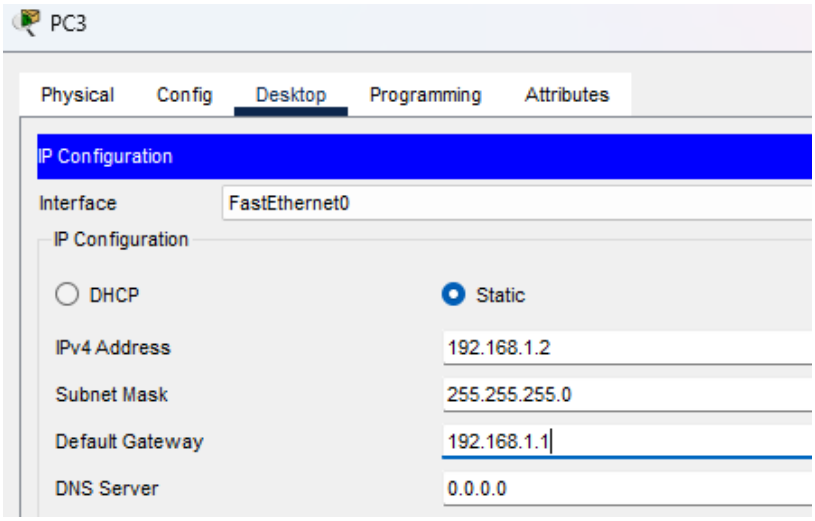
1. Open the packet tracer and implement the following network as a group and make sure to use Activity 1 discussions to configure each device in the network.
   a. You need to make sure PC1, PC3 and PC5 can communicate with each other. Tip: You need to add static routes to each router.
   b. Verify that PC1 can communicate with both PC3 and PC5 using PING command and simulations.



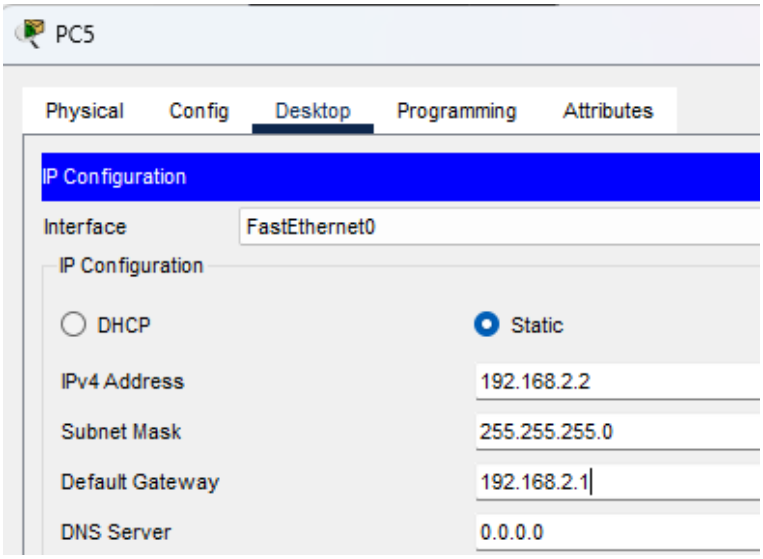
Network Architecture



Configuring IP of PC1



Configuring IP of PC3



Configuring IP of PC5

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface FastEthernet0/0
Router(config-if)#ip address 192.168.3.1 255.255.255.0
Router(config-if)#ip address 192.168.3.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up

Router(config-if)#exit
Router(config)#interface Serial0/0
Router(config-if)#ip address 10.0.0.2 255.0.0.0
Router(config-if)#ip address 10.0.0.2 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state to up

Router(config-if)#exit
Router(config)#
Router(config)#ip route 192.168.1.0 255.255.255.0 10.0.0.3
Router(config)#ip route 192.168.2.0 255.255.255.0 10.0.0.3
Router(config)#ip route 11.0.0.0 255.0.0.0 10.0.0.3
Router(config)#
```

Configuring Router1 by adding IP address to both ethernet and serial ports and setting up a static connection.

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface FastEthernet0/0
Router(config-if)#ip address 192.168.1.1 255.255.255.0
Router(config-if)#ip address 192.168.1.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up

Router(config-if)#exit
Router(config)#interface Serial0/0
Router(config-if)#ip address 10.0.0.2 255.0.0.0
Router(config-if)#ip address 10.0.0.2 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state to up
ip address 10.0.0.3 255.0.0.0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial0/1
Router(config-if)#ip address 11.0.0.2 255.0.0.0
Router(config-if)#ip address 11.0.0.2 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1, changed state to up

Router(config-if)#exit
Router(config)#
Router(config)#ip route 192.168.3.0 255.255.255.0 10.0.0.2
Router(config)#ip route 192.168.2.0 255.255.255.0 11.0.0.3
Router(config)#
```

Configuring Router2 by adding IP address to both ethernet and serial ports and setting up a static connection.

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface FastEthernet0/0
Router(config-if)#ip address 192.168.2.1 255.255.255.0
Router(config-if)#ip address 192.168.2.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up

Router(config-if)#exit
Router(config)#interface Serial0/0
Router(config-if)#ip address 11.0.0.3 255.0.0.0
Router(config-if)#ip address 11.0.0.3 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state to up

Router(config-if)#exit
Router(config)#
Router(config)#ip route 192.168.1.0 255.255.255.0 11.0.0.2
Router(config)#ip route 192.168.3.0 255.255.255.0 10.0.0.2
Router(config)#ip route 192.168.3.0 255.255.255.0 11.0.0.2
Router(config)#no ip route 192.168.3.0 255.255.255.0 10.0.0.2
Router(config)#ip route 10.0.0.0 255.0.0.0 11.0.0.2
Router(config)#
```

Configuring Router3 by adding IP address to both ethernet and serial ports and setting up a static connection.

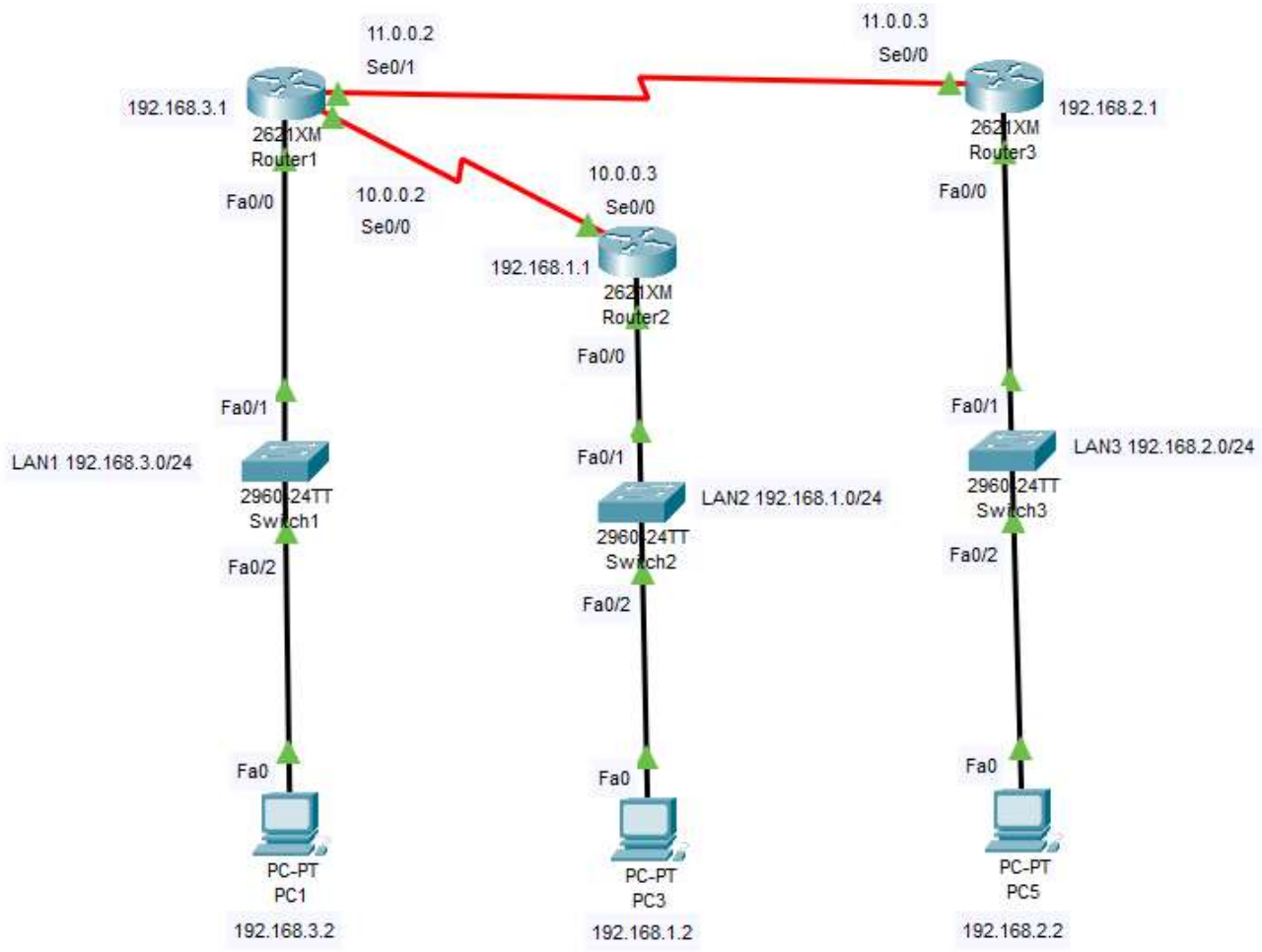
```
C:\>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Reply from 192.168.1.2: bytes=32 time=12ms TTL=126
Reply from 192.168.1.2: bytes=32 time=10ms TTL=126
Reply from 192.168.1.2: bytes=32 time=84ms TTL=126
Reply from 192.168.1.2: bytes=32 time=9ms TTL=126

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 9ms, Maximum = 84ms, Average = 28ms

C:\>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Reply from 192.168.2.2: bytes=32 time=81ms TTL=125
Reply from 192.168.2.2: bytes=32 time=66ms TTL=125
Reply from 192.168.2.2: bytes=32 time=16ms TTL=125
Reply from 192.168.2.2: bytes=32 time=10ms TTL=125

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 10ms, Maximum = 81ms, Average = 43ms
```

Pinging both PC3 or PC5 from PC1 Successfully

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC1	PC3	ICMP		0.000	N	0	(edit)	(delete)
	Successful	PC1	PC5	ICMP		0.000	N	1	(edit)	(delete)
	Successful	PC3	PC1	ICMP		0.000	N	2	(edit)	(delete)
	Successful	PC3	PC5	ICMP		0.000	N	3	(edit)	(delete)

Simulating Packet transfers between ll three PCs successfully

2. Now, add another serial connection between Router 1 and 3 and remove the direct connection between router 2 and router 3.



Updated Network Architecture

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#
Router(config)#
Router(config)#interface Serial0/1
Router(config-if)#ip address 11.0.0.2 255.0.0.0
Router(config-if)#ip address 11.0.0.2 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1, changed state to up

Router(config-if)#exit
Router(config)#
Router(config)#ip route 192.168.2.0 255.255.255.0 11.0.0.3
Router(config)#no ip route 192.168.2.0 255.255.255.0 10.0.0.3
Router(config)#no ip route 11.0.0.0 255.0.0.0 10.0.0.3
Router(config)#
```

Adding the serial connection between Router1 and Router3 and updating the static routes in Router1

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#
Router(config)#no ip route 192.168.2.0 255.255.255.0 11.0.0.3
Router(config)#ip route 192.168.2.0 255.255.255.0 10.0.0.2
Router(config)#ip route 11.0.0.0 255.0.0.0 10.0.0.2
Router(config)#
```

Updating the static routes in Router2

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#
Router(config)#no ip route 192.168.3.0 255.255.255.0 11.0.0.2
Router(config)#no ip route 192.168.1.0 255.255.255.0 11.0.0.2
Router(config)#no ip route 10.0.0.0 255.0.0.0 11.0.0.2
Router(config)#ip route 192.168.3.0 255.255.255.0 11.0.0.2
Router(config)#ip route 192.168.1.0 255.255.255.0 11.0.0.2
Router(config)#ip route 10.0.0.0 255.0.0.0 11.0.0.2
Router(config)#
```

Updating the static routes in Router3

```
C:\>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Reply from 192.168.2.2: bytes=32 time=16ms TTL=126
Reply from 192.168.2.2: bytes=32 time=13ms TTL=126
Reply from 192.168.2.2: bytes=32 time=15ms TTL=126
Reply from 192.168.2.2: bytes=32 time=63ms TTL=126

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 13ms, Maximum = 63ms, Average = 26ms
```

Successfully pinging from PC1 to PC5

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC1	PC5	ICMP		0.000	N	0	(edit)	
	Successful	PC5	PC1	ICMP		0.000	N	1	(edit)	

Successful packet transfer Simulations between PC1 and PC5

Activity 3

1. Is there any alternative to stop manually configuring the routing table every time the connectivity changes?

- An alternative for static routing is dynamic routing. Dynamic routing enables routers to automatically share and update their routing tables.

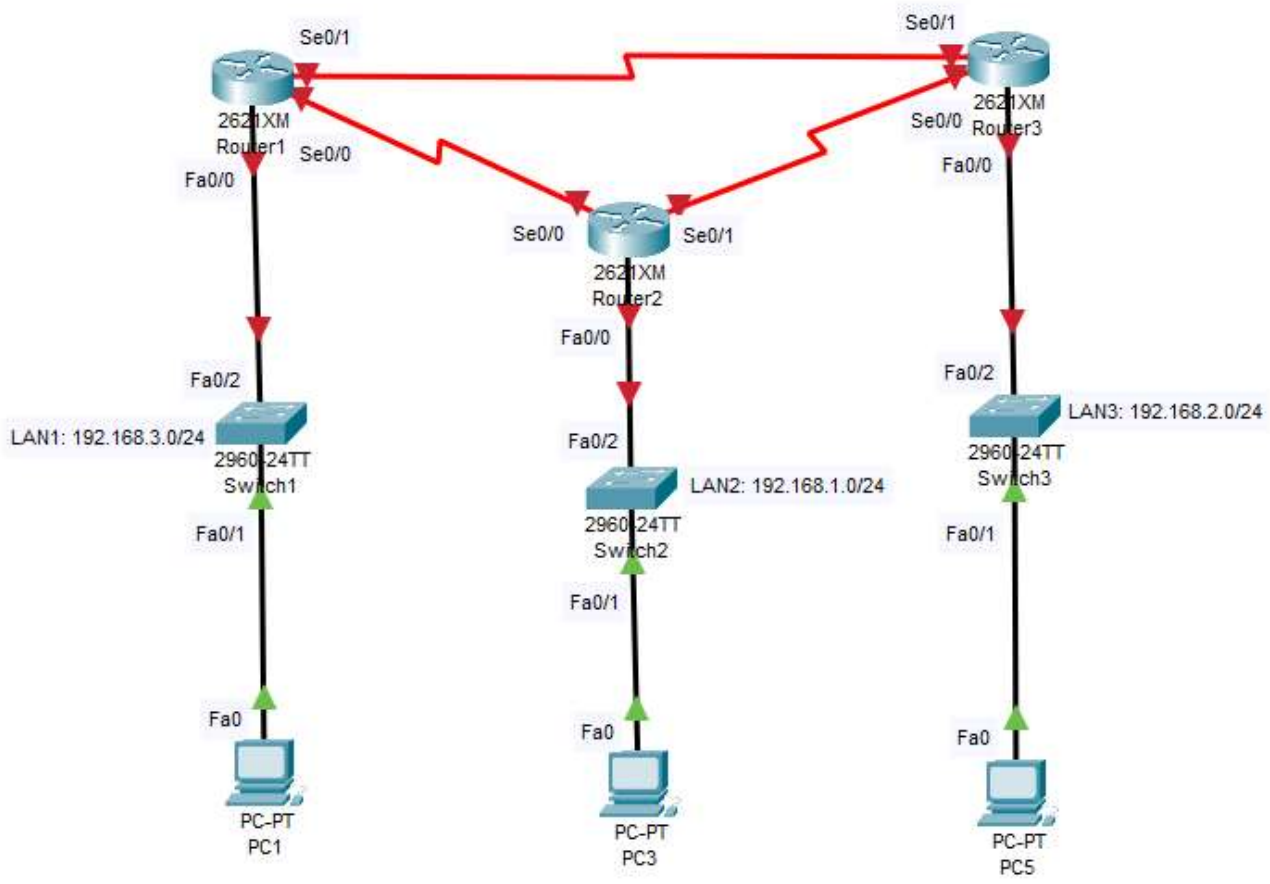
2. What are the latest routing algorithms you can use?

- Software-Defined Networking (SDN) – separates the control plane from the data plane, allowing network administrators to centrally manage routing decisions using software

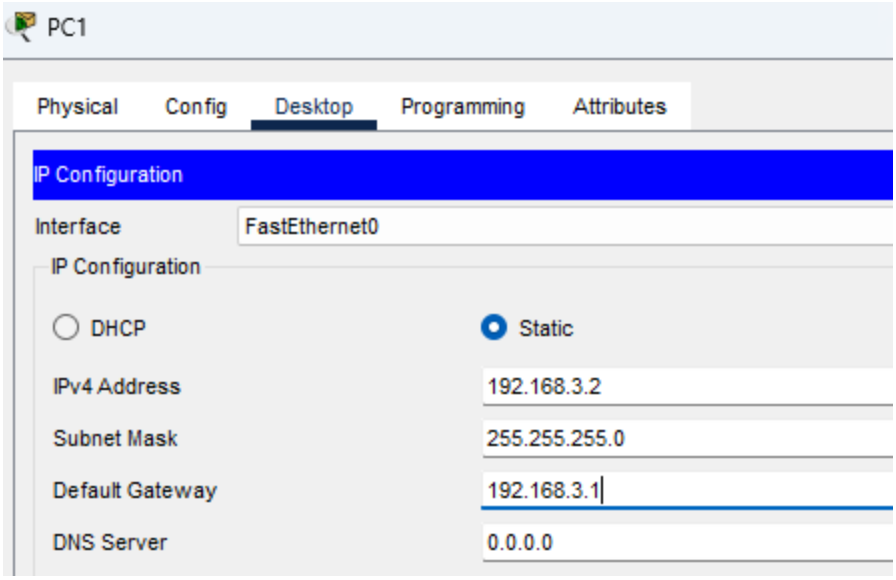
3. Is there any routing algorithm available in Cisco Packet Traces?

- Open Shortest Path First (OSPF)
- Routing Information Protocol (RIP)
- Border Gateway Protocol (BGP)

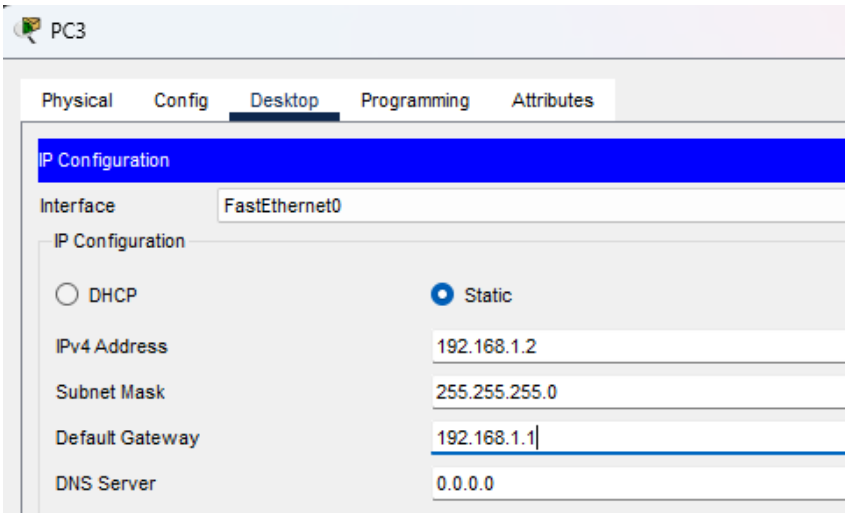
4. Implement the same network as shown in the following figure. However, this time make sure to use the routing algorithm available in the routers.



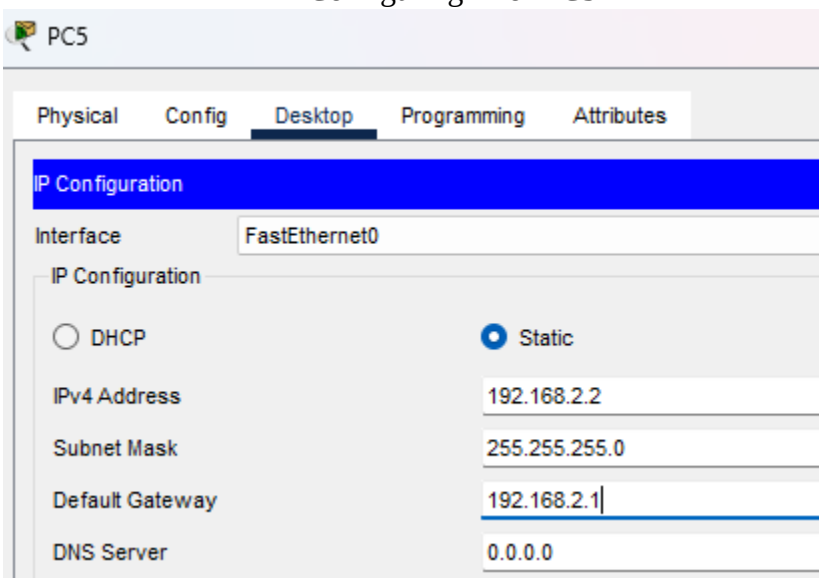
Network Architecture



Configuring IP of PC1



Configuring IP of PC3



Configuring IP of PC5

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface FastEthernet0/0
Router(config-if)#ip address 192.168.3.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up
```

Configuring Router1 Ethernet connection with PC1

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface Serial0/0
Router(config-if)#ip address 10.0.0.2 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#clock rate 64000
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state to up

Router(config-if)#exit
Router(config)#interface Serial0/1
Router(config-if)#clock rate 64000
Router(config-if)#ip address 12.0.0.2 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1, changed state to up

Router(config-if)#exit
Router(config)#router rip
Router(config-router)#network 192.168.3.0
Router(config-router)#network 10.0.0.0
Router(config-router)#network 12.0.0.0
Router(config-router)#
```

Configuring serial connection and Routing Information Protocol networks in Router1

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface FastEthernet0/0
Router(config-if)#ip address 192.168.1.1 255.255.255.0
Router(config-if)#ip address 192.168.1.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up
```

Configuring Router2 Ethernet connection with PC3

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface Serial0/0
Router(config-if)#clock rate 64000
This command applies only to DCE interfaces
Router(config-if)#ip address 10.0.0.3 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/0, changed state to up

Router(config-if)#exit
Router(config)#interface Serial0/1
Router(config-if)#clock rate 64000
Router(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state to up
ip address 11.0.0.2 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1, changed state to up

Router(config-if)#exit
Router(config)#router rip
Router(config-router)#network 192.168.1.0
Router(config-router)#network 10.0.0.0
Router(config-router)#network 11.0.0.0
Router(config-router)#
```

Configuring serial connection and Routing Information Protocol networks in Router2

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface FastEthernet0/0
Router(config-if)#ip address 192.168.2.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up
```

Configuring Router3 Ethernet connection with PC5

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface Serial0/0
Router(config-if)#clock rate 64000
This command applies only to DCE interfaces
Router(config-if)#ip address 11.0.0.3 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state to up

Router(config-if)#exit
Router(config)#interface Serial0/1
Router(config-if)#clock rate 64000
This command applies only to DCE interfaces
Router(config-if)#ip address 12.0.0.3 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1, changed state to up

Router(config-if)#exit
Router(config)#router rip
Router(config-router)#network 192.168.2.0
Router(config-router)#network 11.0.0.0
Router(config-router)#network 12.0.0.0
Router(config-router)#
```

Configuring serial connection and Routing Information Protocol networks in Router3

5. Verify the network connections using PING (in each PC).

```
C:\>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Reply from 192.168.1.2: bytes=32 time=73ms TTL=126
Reply from 192.168.1.2: bytes=32 time=2ms TTL=126
Reply from 192.168.1.2: bytes=32 time=2ms TTL=126
Reply from 192.168.1.2: bytes=32 time=13ms TTL=126

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 73ms, Average = 22ms

C:\>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Reply from 192.168.2.2: bytes=32 time=16ms TTL=126
Reply from 192.168.2.2: bytes=32 time=72ms TTL=126
Reply from 192.168.2.2: bytes=32 time=17ms TTL=126
Reply from 192.168.2.2: bytes=32 time=19ms TTL=126

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 16ms, Maximum = 72ms, Average = 31ms
```

Successfully Pinging PC3 and PC5 from PC1

```
C:\>ping 192.168.3.2

Pinging 192.168.3.2 with 32 bytes of data:

Reply from 192.168.3.2: bytes=32 time=20ms TTL=126
Reply from 192.168.3.2: bytes=32 time=9ms TTL=126
Reply from 192.168.3.2: bytes=32 time=9ms TTL=126
Reply from 192.168.3.2: bytes=32 time=8ms TTL=126

Ping statistics for 192.168.3.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 8ms, Maximum = 20ms, Average = 11ms

C:\>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Reply from 192.168.2.2: bytes=32 time=80ms TTL=126
Reply from 192.168.2.2: bytes=32 time=19ms TTL=126
Reply from 192.168.2.2: bytes=32 time=13ms TTL=126
Reply from 192.168.2.2: bytes=32 time=27ms TTL=126

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 13ms, Maximum = 80ms, Average = 34ms
```

Successfully Pinging PC1 and PC5 from PC3

```
C:\>ping 192.168.3.2

Pinging 192.168.3.2 with 32 bytes of data:

Reply from 192.168.3.2: bytes=32 time=15ms TTL=126
Reply from 192.168.3.2: bytes=32 time=78ms TTL=126
Reply from 192.168.3.2: bytes=32 time=13ms TTL=126
Reply from 192.168.3.2: bytes=32 time=14ms TTL=126

Ping statistics for 192.168.3.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 13ms, Maximum = 78ms, Average = 30ms

C:\>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Reply from 192.168.1.2: bytes=32 time=82ms TTL=126
Reply from 192.168.1.2: bytes=32 time=17ms TTL=126
Reply from 192.168.1.2: bytes=32 time=13ms TTL=126
Reply from 192.168.1.2: bytes=32 time=14ms TTL=126

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 13ms, Maximum = 82ms, Average = 31ms
```

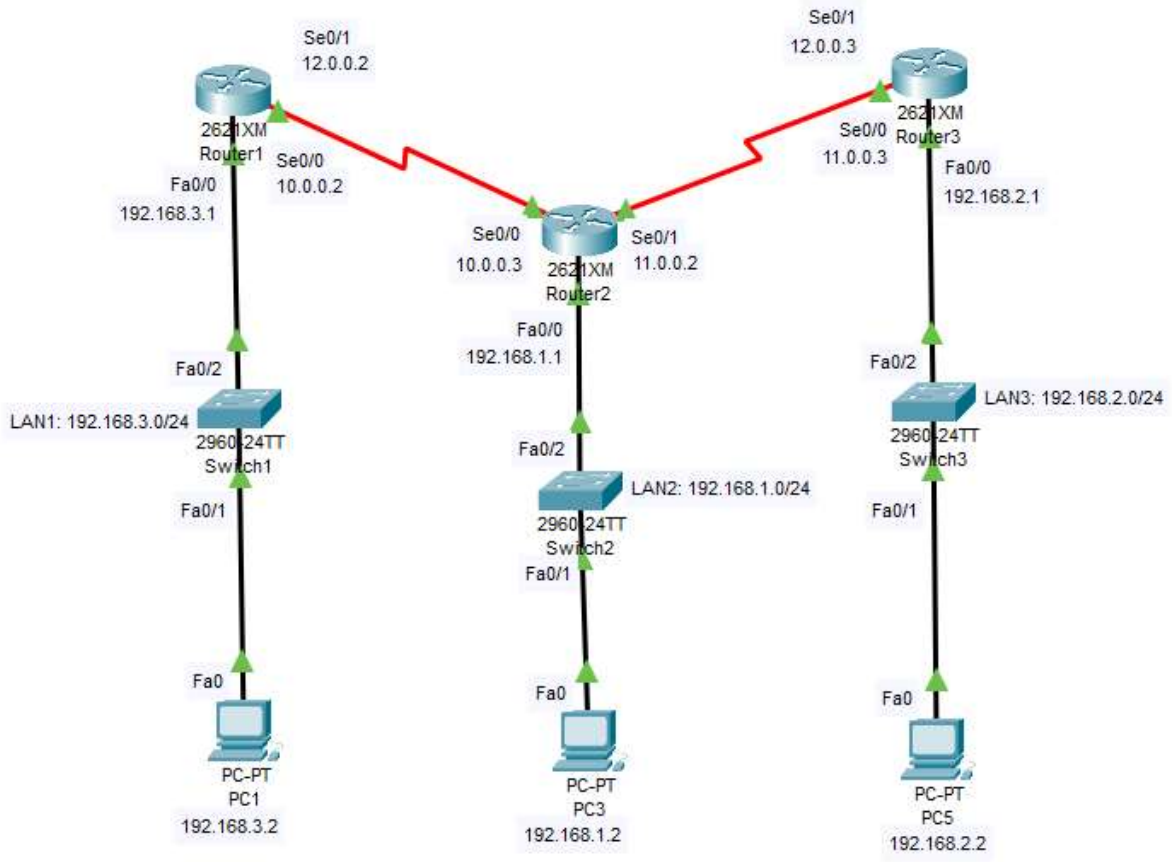
Successfully pinging PC1 and PC3 from PC5

6. Use the simulation tool to send a packet from PC1 to PC5. Have you seen any differences compared to the route you have seen in Activity 2? Explain what happened.

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC1	PC5	ICMP		0.000	N	0	(edit)	(delete)

Instead of using Static Routing, Dynamic Routing was implemented. The protocol followed specifically was Routing Information Protocol.

7. Now remove the connection between router 1 and 3 and use the simulation tool to send a packet from PC1 to PC5 without changing any configuration.



Updated Networking system

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete	
	Successful	PC1	PC5	ICMP		0.000	N	0	(edit)		(delete)
	Successful	PC1	PC5	ICMP		0.000	N	1	(edit)		(delete)

Successful packet transfer from PC1 to PC5

8. Explain what you observed.
    - Within a network that is connected, the Routing Information Protocol (RIP) sets up routers to determine the best route for sending data packets from a sender to a recipient. In this instance, since Router 1 and Router 3's serial connection was severed. The route from Router 1 to Router 2 to Router 3 is the next best one with the fewest hops that can be made.

Above and Beyond Tasks

1. **Routing Information Protocol (RIP)**
   - **Type**: Distance Vector
   - **Algorithm**: Uses the Bellman-Ford algorithm.
   - **Metric**: Hop count, with a maximum limit of 15 hops.
   - **Updates**: Periodic updates every 30 seconds.
   - **Convergence Time**: Slow convergence due to periodic updates and limited metric (hop count).
   - **Scalability**: Suitable for small networks due to the hop count limit.

     Advantages:
     - Simple to configure and easy to implement.
     - Low CPU/memory overhead due to minimal processing.

     Disadvantages:
     - Poor scalability (limited to 15 hops).
     - Slow convergence and prone to routing loops.

2. **Open Shortest Path First (OSPF)**
   - **Type**: Link State
   - **Algorithm**: Uses Dijkstra's Shortest Path First (SPF) algorithm.
   - **Metric**: Cost (based on bandwidth), making it more efficient for varying link speeds.
   - **Updates**: Event-driven updates (only when changes occur), reducing bandwidth usage.
   - **Convergence Time**: Fast convergence, as changes are rapidly propagated.
   - **Scalability**: Highly scalable, suitable for both small and large networks. Supports hierarchical design with areas.

     Advantages:
     - Fast convergence and better for large, complex networks.
     - Supports multi-area network structures for efficient routing.

     Disadvantages:
     - More complex to configure and manage compared to RIP.
     - Higher CPU and memory overhead due to its algorithm and state tracking.

3. **Border Gateway Protocol (BGP)**
   - **Type**: Path Vector
   - **Algorithm**: Uses path attributes to determine the best route.
   - **Metric**: Uses multiple attributes like AS path, next-hop, and policy-based metrics.
   - **Updates**: Event-driven updates (only when network topology changes).
   - **Convergence Time**: Slower convergence compared to OSPF due to its complexity.
   - **Scalability**: Highly scalable, used for inter-domain (between ISPs) and large networks like the Internet.

     Advantages:
     - Essential for Internet backbone routing and inter-domain communication.
     - Highly customizable via route policies.

     Disadvantages:
     - Very complex to configure and maintain.
     - Slow convergence due to complex decision processes and large networks.

SIT202: Computer Networks and Communication
Task 7.1P Learning Plan Check-in

Name: Kenisha Corera
Student ID: DFCS|DK|62|203

Reflection on Modules so far…

My understanding of all things network-related has grown thanks in large part to the Computer Networks and Communication Module thus far. Network system creation in Cisco Packet Tracer, packet sniffing in Wireshark, and the interaction between each layer in the TCP/IP paradigm are all examples of this. I learnt new topics like socket programming from this model, which truly grabbed my attention, in addition to reviewing topics I already understood. I'll keep concentrating on my objective, which is to finish every assignment and aspire for a Higher Distinction.

Tasks	Due Date (P)	Due Date (C)	Due Date (D)	Due Date (HD)
P tasks (Lesson Reviews)				
Task 1.2P Introduction	Completed			
Task 3.1P Application Layer	Completed			
Task 5.1P Transport Layer	Completed			
Task 6.1P Network Layer-Data Plane	Completed			
Task 8.1P Network Layer – Control Plane	Yet to complete			
Task 9.1P Data Link Layer	Yet to complete			
Task 10.1P Physical Layer	Yet to complete			
C tasks				
Task 4.1 C Above and Beyond Pass		Yet to complete		
Task 5.2 C My DNS Server Sketch		Completed		
Task 6.2 C My DNS Server		Completed		
D tasks				
Task 5.2 D Above and Beyond Credit			Yet to complete	
Task 7.2D Amazing Networks			Yet to complete	

HD tasks				
Task 1.3 H Collaborative Learning Initiatives & Outcomes (aim for exemplary achievement 90+)				Yet to complete
Task 6.3 H My Awesome Tutorial				Yet to complete
Task 8.2 H Building Your Own Ping Program				Yet to complete

Computer Networks and Communication

Module Summary

This subject covered a number of important topics related to computer networks, such as the allocation and maintenance of IP addresses and the use of fundamental network protocols. Dealing with the global IPv4 address shortage—a problem caused by the small amount of IP addresses available in the IPv4 system—was one of the main tasks. We covered how to assign IP addresses to devices automatically using the Dynamic Host Configuration Protocol (DHCP), making the best use of the remaining address space, and how to use Network Address Translation (NAT) to allow multiple devices on a local network to share a single public IP address.

Using Cisco Packet Tracer, we had to create a network, which included setting up a DHCP server, defining network pools, and making ensuring that hosts were connected to one another. This exercise taught us how to set up a router to dynamically assign IP addresses to subnets, manage numerous subnets, and use various network tools to confirm connections. To further understand how data is transmitted over the network, we also used the ping and traceroute commands to investigate the Internet Control Message Protocol (ICMP) and analyse the packet information.

An further beneficial activity involved recording and examining DHCP packets using Wireshark. This enabled us to view the real flow of communications, including DHCP find, offer, request, and acknowledgement messages, that transpire between devices and the DHCP server. Additionally, we investigated the distinctions between intra-AS (Autonomous System) and inter-AS routing protocols. We discovered that while inter-AS protocols, such as BGP (Border Gateway Protocol), handle routing between multiple ASes and prioritise scalability, intra-AS protocols, such as OSPF (Open Shortest Path First), concentrate on performance within a single AS.

Reflecting on the content

I gained useful knowledge from this module that I can immediately use to networking situations in the real world. Using DHCP to configure a network showed how important automation is for managing IP addresses, a crucial task for large-scale networks. I had a primarily theoretical understanding of IP addressing and network configuration before taking this module. My understanding of how protocols like DHCP and NAT cooperate to guarantee effective network functioning has improved as a result of the practical exercises.

Analyzing ICMP packets and comprehending how devices communicate through various network tiers was one of the most crucial lessons learnt. My understanding of networks' layered design and how various protocols manage particular communication-related jobs has improved as a result of this.

By combining theoretical understanding with real-world application, I think the course designers aimed to provide us a comprehensive understanding of network configuration and protocol analysis. With this strategy, we can be confident that we are ready to take on real-world problems like managing networks with a restricted number of IP addresses and guaranteeing smooth device-to-device communication. My technical proficiency as well as my ability to successfully analyse and troubleshoot network setups have both increased as a result of this training.

SIT202: Computer Networks and Communication

Leaning Evidence for Active Class Task 8

Name: Kenisha Corera
Student ID: C23020001

Members in this group activity task:

Nishad – C23110001
Kenisha – C23020001
Shekaina – C23110002
Raaid – C23020004
Pavithran – C22060015

Activity 1

1. Do we have enough IP addresses to assign for each device that connect to the network?

- IP addresses, especially IPv4, are hard to come by, especially on public networks. With 32 bits in a conventional IPv4 address, there are roughly 4.3 billion possible unique addresses. However, because of network allocation policies and the separation of addresses into public and private sectors, this pool is progressively diminished. You should have enough private IP addresses, if you're utilizing them, to cover most local networks. However, because the IPv4 pool is being used up, there are far fewer unique IPv4 addresses available on public networks.

2. Do we have any solution?

- To cope with IPv4 addresses being scarce. A number of actions are taken. o Network Address Translation: When devices in a local network use private IP addresses, a router or firewall translates these private addresses into a single public IP address for communication on the public network. This is a common method of conserving public IP addresses. By enabling numerous devices to share a single public IP address, IPv4 IP addresses can be used much more efficiently.

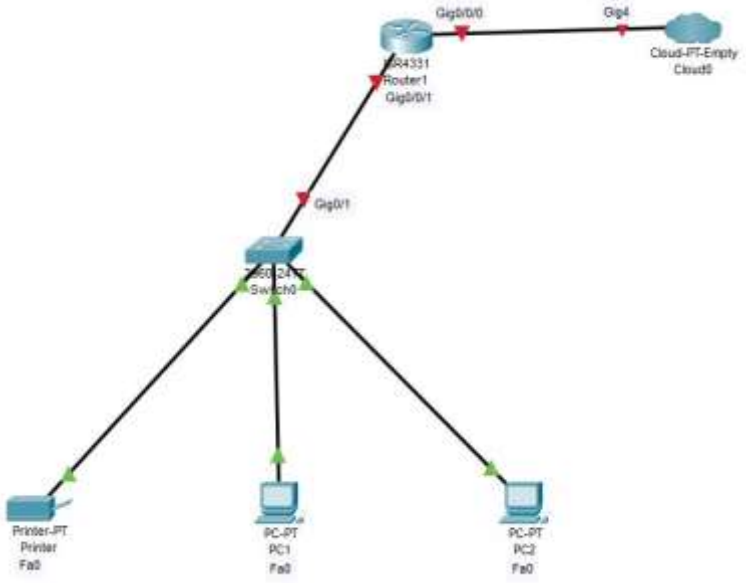
3. Explain a protocol that we can use along with IPv4 to conserve the global IP address space?

- We can use the Dynamic Host Configuration Protocol (DHCP) with IPV4. In order to prevent IP addresses from being squandered, DHCP automatically allocates available IP addresses to connected devices.

.

Activity 2

1. One group member can act as a router with a DHCP server and other three members are host devices that try to connect to the network. These host devices could be a PC, a printer, and network storage device. A sample network is shown in the below figure.



2. Assume the all the devices are physical connected as shown in the above figure. Now, your job is to set up the network and establish the connections with the router using DHCP. Each member needs to send/ respond with the correct message and message sequence to establish the connections and receive IP address for each host.

Pavithran AIR
Pavithran
"Welcome to our DHCP network setup roleplay! In this scenario, we have a Router acting as the DHCP Server and three host devices — PC 1, PC 2, and Printer — that need to connect to the network. Let's get started!"
9:12 AM

PC 1 – Raaid
"I am PC 1. I want to join the network, but I need an IP address first."
(Speaks to Router) "Sending DHCP Discover message to the network. Who can provide me an IP address?"
9:15 AM

Shekaina CICRA
PC 2 – Shekaina
(Follows the same action) "I am PC 2, and I also need an IP address."
(Sends DHCP Discover message) "Who is out there to provide me with an IP address?"
Edited 9:18 AM

Nishad
Printer – Nishad
(Receives the IP Address card and speaks) "I have my IP address offer. I am confirming it now!"
(Sends DHCP Request message to the Router) "Sending DHCP Request message to confirm the IP address."
9:19 AM

Kenisha CICRA
Router/DHCP Server- kenisha
"I am the Router, acting as the DHCP Server. I have received your DHCP Discover messages."
"Sending DHCP Offer messages to PC 1, PC 2, and Printer with an available IP address for each of you."
"Sending DHCP Acknowledgment (ACK) to confirm that each device now has an IP address assigned."
(Hands out "ACK cards" to each host device member)
Edited 9:20 AM

PC 1 – Raaid
(Receives the IP Address card and speaks) "I received an IP address offer. I accept this IP address!"
(Sends DHCP Request message to the Router) "Sending DHCP Request message to confirm the IP address."
9:22 AM

Shekaina CICRA
PC 2 – Shekaina
(Receives the IP Address card and speaks) "I also received an IP address offer. I am confirming my IP address as well!"
(Sends DHCP Request message to the Router) "Sending DHCP Request message to confirm the IP address."
9:31 AM

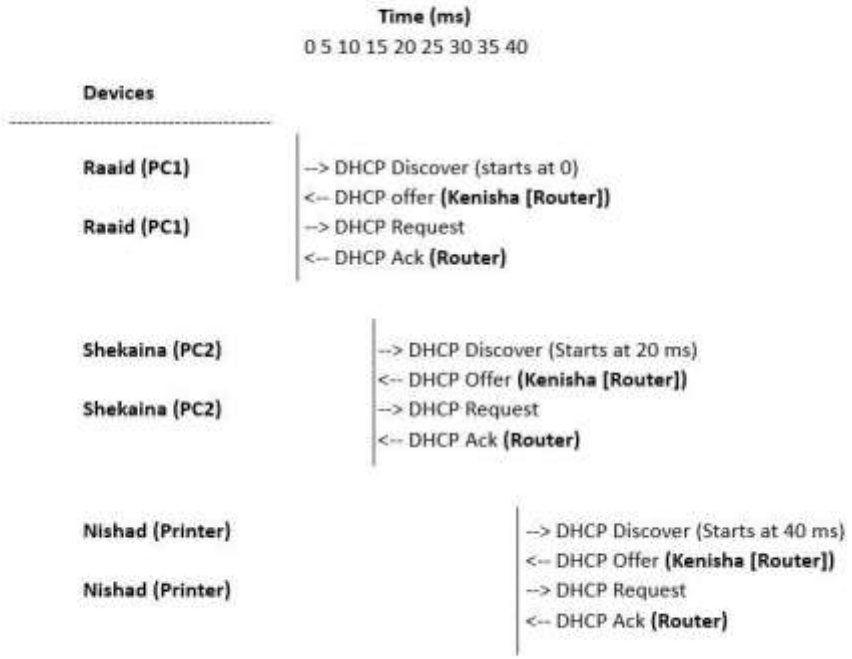
Nishad
Printer – Nishad
(Receives the IP Address card and speaks) "I have my IP address offer. I am confirming it now!"
(Sends DHCP Request message to the Router) "Sending DHCP Request message to confirm the IP address."
9:47 AM

Kenisha CICRA
Router/DHCP Server- kenisha
(Receives the DHCP Request messages) "I have received the DHCP Requests from PC 1, PC 2, and Printer."
"Sending DHCP Acknowledgment (ACK) to confirm that each device now has an IP address assigned."
(Hands out "ACK cards" to each host device member)
PC 1 – Raaid , PC 2 – Shekaina , Printer – Nishad
(Receive the ACK cards) "We have received the DHCP ACK! We are now connected to the network with our IP addresses."
9:58 AM

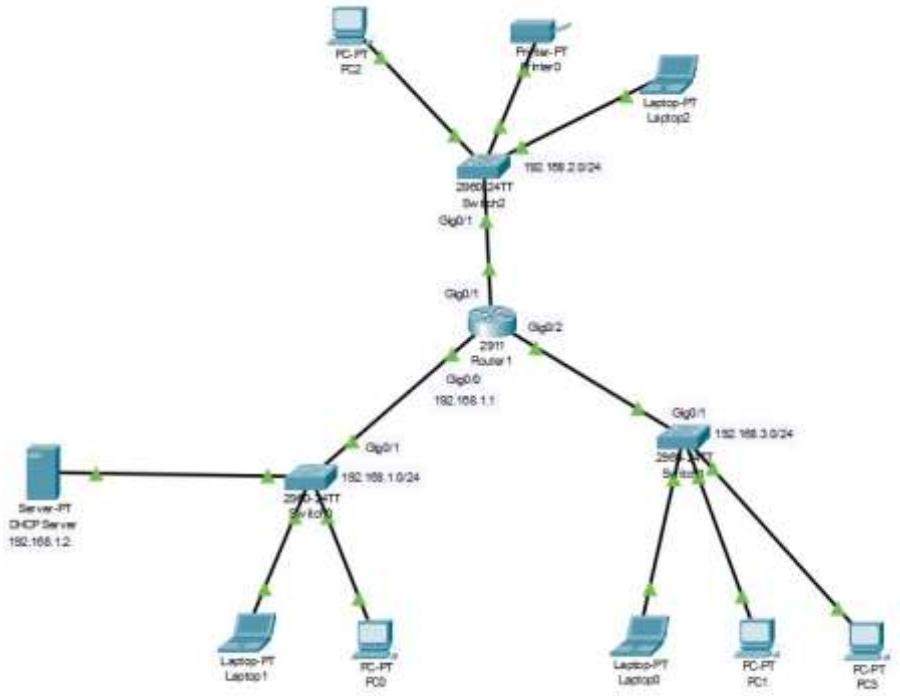
Pavithran AIR
Pavithran
"Great job, team! All devices have successfully received their IP addresses from the DHCP Server and are now connected to the network. Our DHCP setup is complete!"
9:58 AM

3. Draw a timing diagram to indicate the sequence of messages transferred between devices.

Time line diagram



4. Next, we are going to implement a DHCP server in Cisco packet tracer and check the DHCP in action. Make use to take screenshots of the networks you build, analysis, and verifications. Open the packet tracer and implement the following network. You may need to use the same model of the devices shown in the diagram. Today, we are not going to use static IP configuration for the hosts as we are going to check the DHCP in action. Tips: You need to add a module with a fast ethernet port to the router, so that the router can support three LANs as shown in the below network diagram.



Network architecture

a. Set up the DHCP server by configuring network pools, default gateway, and IP address (use the subnets mentioned in the diagram).
b. Set up the router by configuring the interfaces and DHCP (use the subnets mentioned in the diagram).
c. Set up each host with DHCP to obtain IP configuration and verify the IP address and the default gateway configured.
d. Verify the connectivity between each host in the network.

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface GigabitEthernet0/0
Router(config-if)#ip address 192.168.1.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0, changed state to up

Router(config-if)#exit
Router(config)#interface GigabitEthernet0/1
Router(config-if)#ip address 192.168.2.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state to up

Router(config-if)#exit
Router(config)#interface GigabitEthernet0/2
Router(config-if)#ip address 192.168.3.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/2, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/2, changed state to up
```

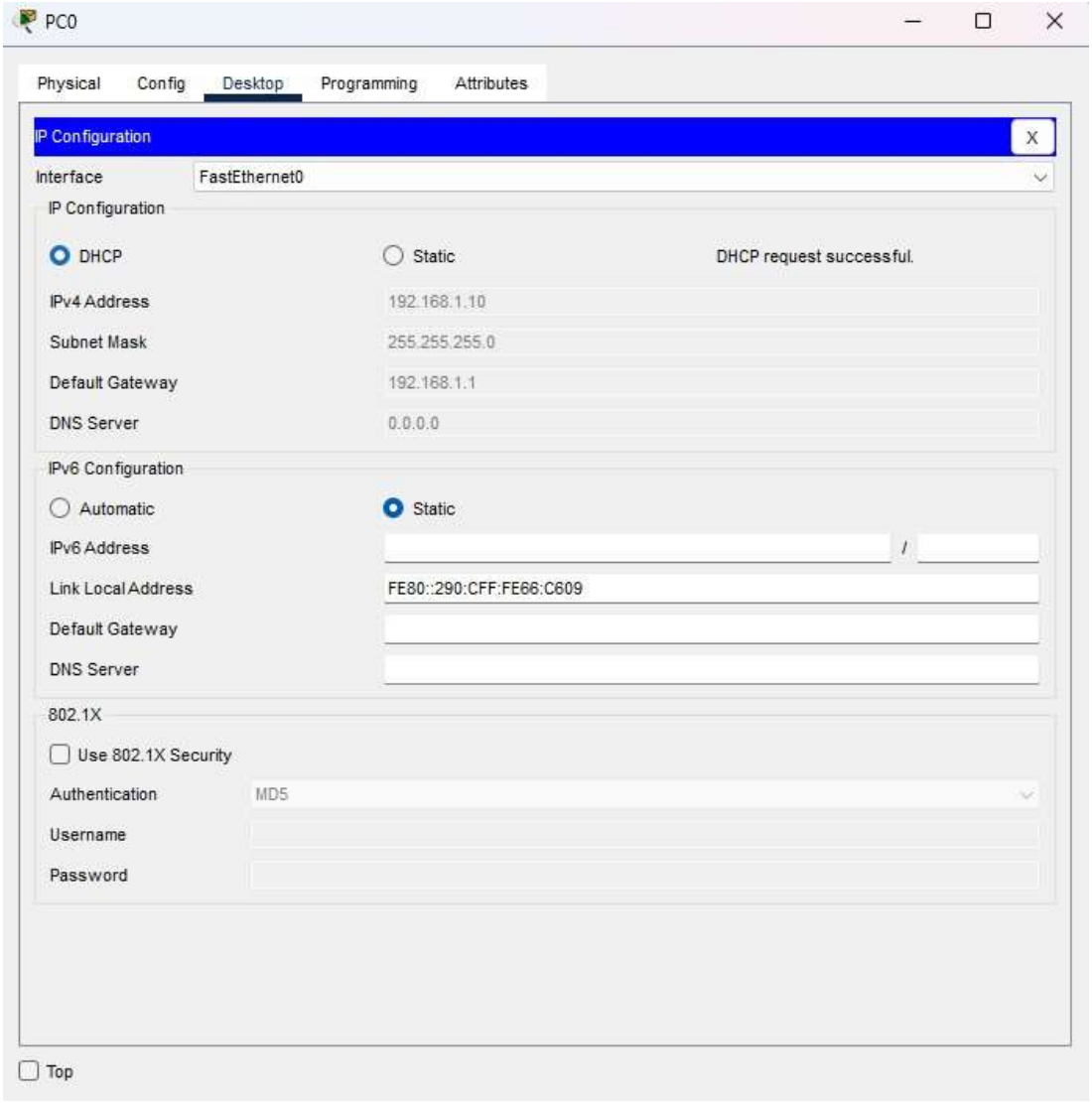
Router IP Configuration

Pool Name	Default Gateway	DNS Server	Start IP Address	Subnet Mask	Max User	TFTP Server	WLC Address
serverPool2	192.168.3.1	0.0.0.0	192.168.3.10	255.255.255.0	100	0.0.0.0	0.0.0.0
serverPool1	192.168.2.1	0.0.0.0	192.168.2.10	255.255.255.0	100	0.0.0.0	0.0.0.0
serverPool	192.168.1.1	0.0.0.0	192.168.1.10	255.255.255.0	100	0.0.0.0	0.0.0.0

Creating DHCP Pools for each subnet

```
Router#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#int g0/1
Router(config-if)#ip helpe
Router(config-if)#ip helper-address 192.168.1.2
Router(config-if)#exit
Router(config)#int g0/2
Router(config-if)#ip helper-address 192.168.1.2
Router(config-if)#exit
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console

Router#wr
Building configuration...
[OK]
```

Enable IP helper address for the other two addresses to enable end device broadcasting to DHCP server



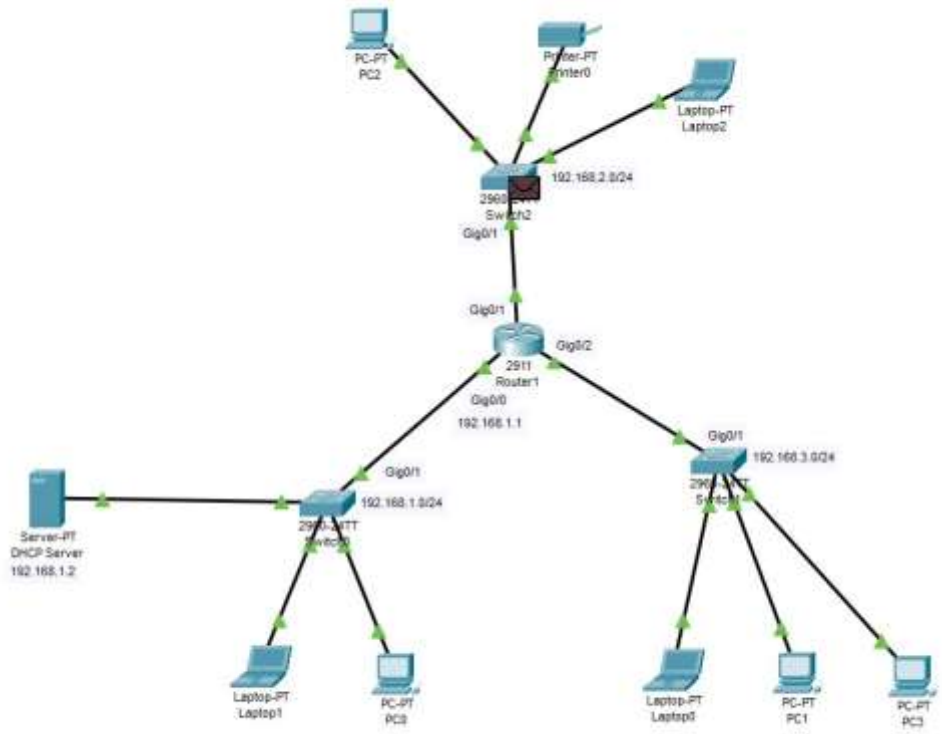
Now each end device as seen above with PC0 can be given an IP address from the DHCP pool.

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	Laptop0	PC2	ICMP		0.000	N	0	(edit)	(delete)
	Successful	Printer0	PC0	ICMP		0.000	N	1	(edit)	(delete)
	Successful	Laptop2	PC3	ICMP		0.000	N	2	(edit)	(delete)
	Successful	Laptop0	Laptop1	ICMP		0.000	N	3	(edit)	(delete)

Thus, we can say that DHCP has been configured and assigned successfully.

Activity 3

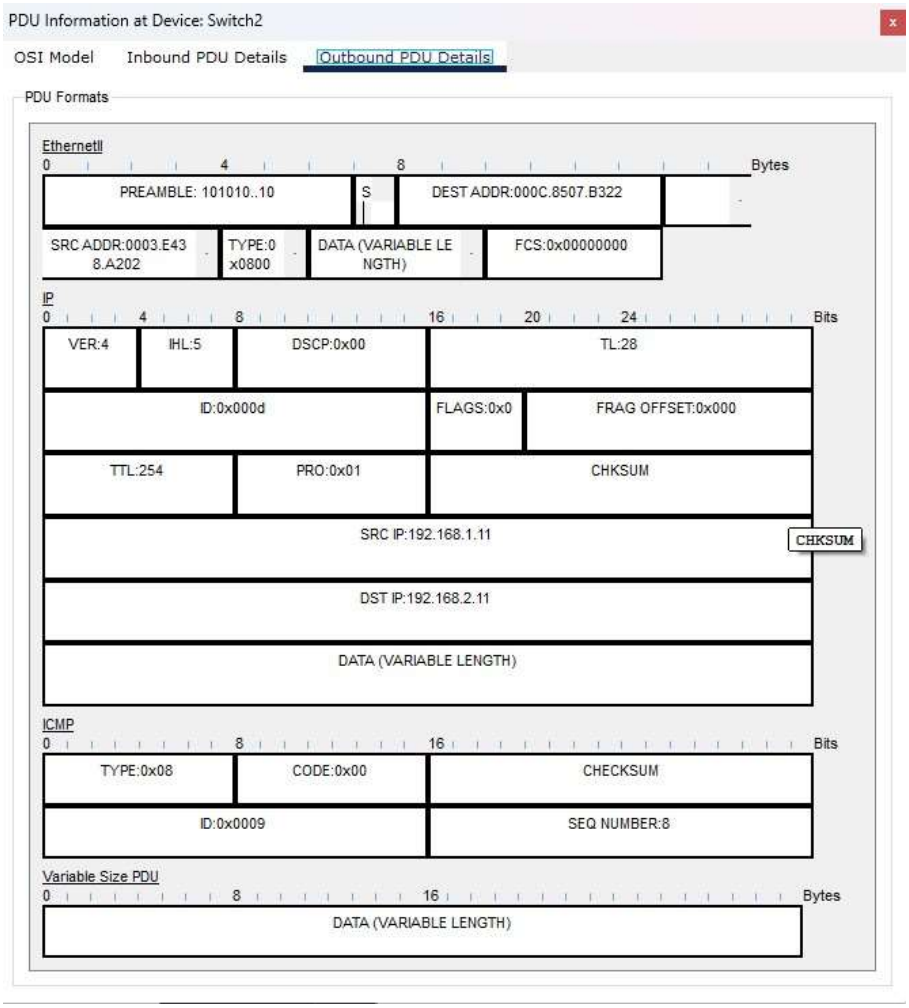
1. Use the Simulation mode and send a simple packet from one host (Host A) from LAN1 to another host (Host B) in LAN2. In simulation tool, you can run the simulation step by step. When you go through each step, pay attention to the type of the message passed (highlighted in the below figure).



- Simulating a ping from Laptop0 to Laptop2

2. You may notice when Host A ping Host B, we use ICMP protocol. Double click on one entry as shown below to explore the ICMP message,

3. You should be able to view the packet details as follows.



4. Explain what information you can find in “outbound PDU details”.

- Outbound PDU details displays the packet’s structure and intricacies on each network layer. As you can see it shows information such as; ○ Destination MAC Address ○ Source MAC Address
    - IP version ○ Header Length ○ Protocol ○ Source IP ○ Destination Ip
    - And ICMP in the Transport Layer

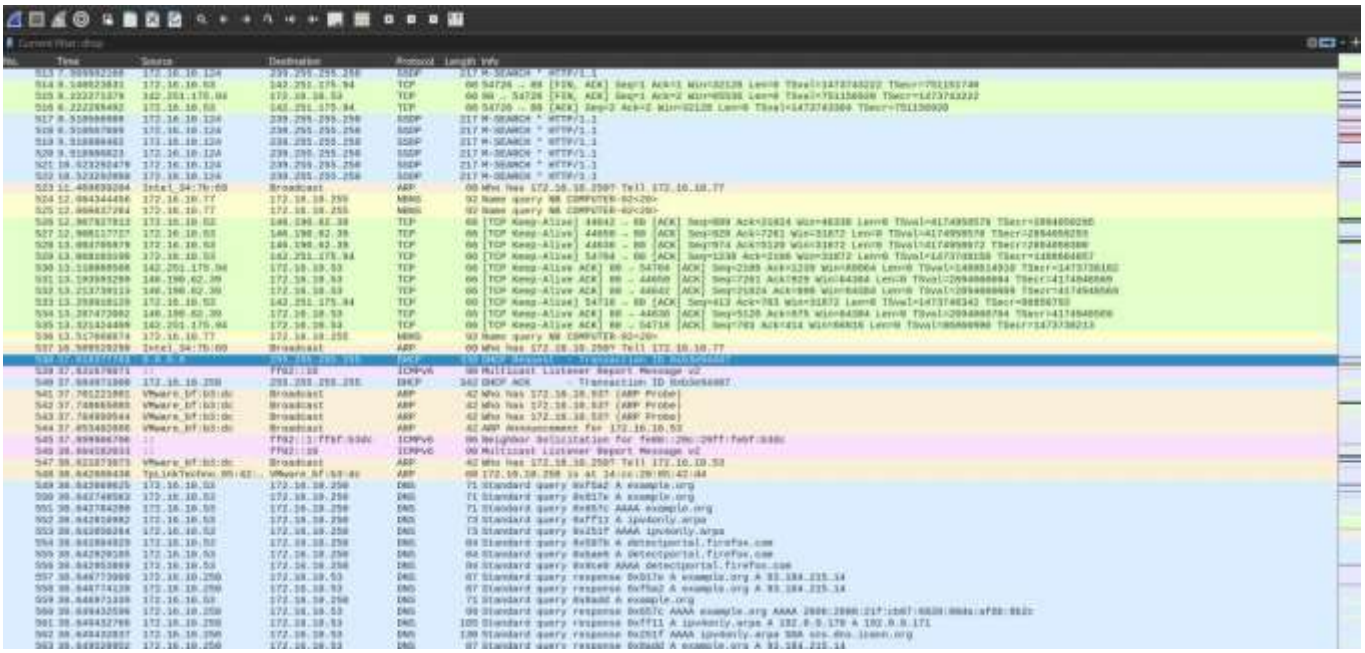
5. Have a closer look at the ICMP message format. Can you identify the type of ICMP message and specific information is in there (type and code)? You can refer to RFC792 for more help. https://datatracker.ietf.org/doc/html/rfc792
    - The type of ICMP message is **0x08** which is a ping (ECHO request) and its ID is **0x0009** and sequence number is 9

6. Can you identify the changes in ICMP message when you use a traceroute from Host A to Host B compared to what you have witnessed in the above question? Note that you can use tracert command in each device (in command prompt) similar to the way that ping command is used.

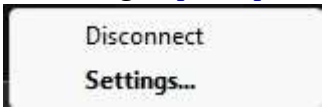
   • The sequence numbers and IDs differ, indicating that the Time to Live (TTL) setting is causing a packet to be discarded after a predetermined amount of time and sending a new one to the next hop.

Above and Beyond Tasks

1. Analyze DHCP in Wireshark (you may want to disconnect your wireless connection and connect it again whilst capturing the packets). You can show the sequence of DHCP messages and the details of those messages.

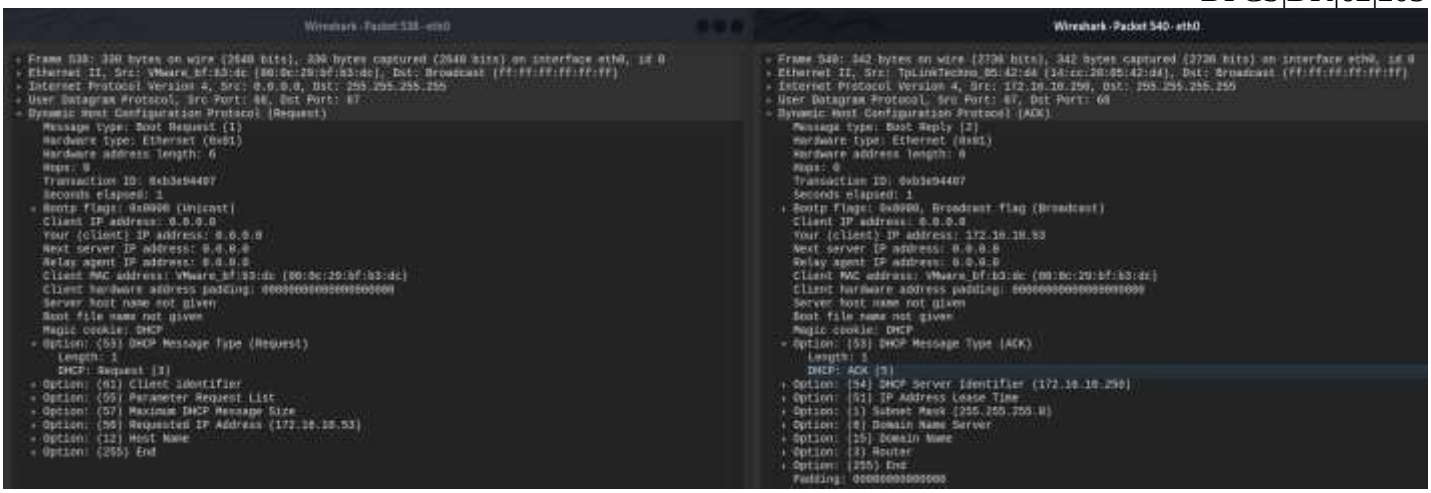


- Enabling Packet Capturing and visiting http://httpforever.com



- Disabling and then re-enabling the Network Adapter in Kali VM to disrupt packet capturing

- Filtering Wireshark by dhcp and as you can see the sequence of DHCP messages with the DHCP Request and DHCP ACK reply.



These are the request and reply packet frames.

2. Why are different inter-AS and intra-AS protocols used in the Internet? What are the examples of these two different types of routing protocols?

- Intra-AS protocols within a single AS whereas Inter-AS protocols handle routing between Aces
- Intra-AS protocols prioritize performance and efficiency while Inter-AS protocols focus on scalability
- An example scenario for Intra-AS could be OSPF where it’s used to efficiently route packets between routers in the same AS. And for Inter-AS BGP is a valid example where its used to manage packets routed across the internet.

Computer Networks and Communication

Module Summary

We covered a wide range of topics related to network communication in this module, with a focus on the function of MAC protocols in Wi-Fi networks and address resolution in IP-based networks. To mimic the functioning of a wireless LAN (WLAN), which consists of a wireless access point and multiple linked devices, including laptops and smartwatches, the group performed a role-play. The focus was on how these devices interact with one another over a shared wireless medium and how the Carrier Sense numerous Access with Collision Avoidance (CSMA/CA) protocol is used to control collisions, which happen when numerous devices attempt to transmit data at the same time. Through the optimal sharing of the wireless medium, this protocol reduces collisions and enhances overall performance among devices.

Also, we looked at the Address Resolution Protocol (ARP), which maps IP addresses to MAC addresses and is essential to network communication. We saw how ARP broadcasts let devices find each other's MAC addresses. We saw this process occurring between devices on the network through hands-on exercises in Cisco Packet Tracer. Our comprehension of how network devices store and utilize these mappings for effective communication has improved as a result of our analysis of the ARP tables in different devices.

The session also examined how switches maintain and store MAC addresses in their tables and how network device communication updates this data. By setting up a DHCP server to dynamically assign IP addresses to devices, we were able to examine how automated IP allocation can streamline network administration and further expand our understanding of this concept.

The final activity covered the security ramifications of ARP, namely flaws like ARP spoofing and the possibility of attacks like denial-of-service (DoS) and man-in-the-middle (MITM). In connection to ARP, this introduced the idea of network security and the necessity of mitigating measures for safeguarding network communication.

Reflecting on the content

My comprehension of network device coordination and communication in wired and wireless environments has been much enhanced by this program. My knowledge of network protocols was primarily theoretical before this assignment, however the practical exercises using ARP and CSMA/CA solidified how these protocols are used in everyday situations. Understanding how network devices detect collisions and recover from them was made easier by the role-play and simulations, which are especially crucial in shared wireless environments like Wi-Fi.

An enlightening glimpse into the fundamental mechanisms underlying device communication on local networks was given by the analysis of ARP. My understanding of how apparently straightforward protocols serve as the foundation of contemporary networking has grown as a result of witnessing how ARP functions at the network layer and how it dynamically creates tables to promote effective communication. The necessity of putting safe networking techniques into place is further highlighted by the examination of ARP's security flaws, which provided a critical viewpoint on how attackers can take advantage of these defences.

We were able to experience the inner workings of network protocols while also learning to recognize and fix potential vulnerabilities in network architecture because this activity was clearly meant to mix theoretical knowledge with practical application. The goal of the course team was probably to give us the knowledge and abilities needed to manage the difficulties of real-world network communication, from setting up devices to identifying and reducing security threats.

SIT202: Computer Networks and Communication
Leaning Evidence for Active Class Task 9

Name: Kenisha Corera
Student ID: C23020001

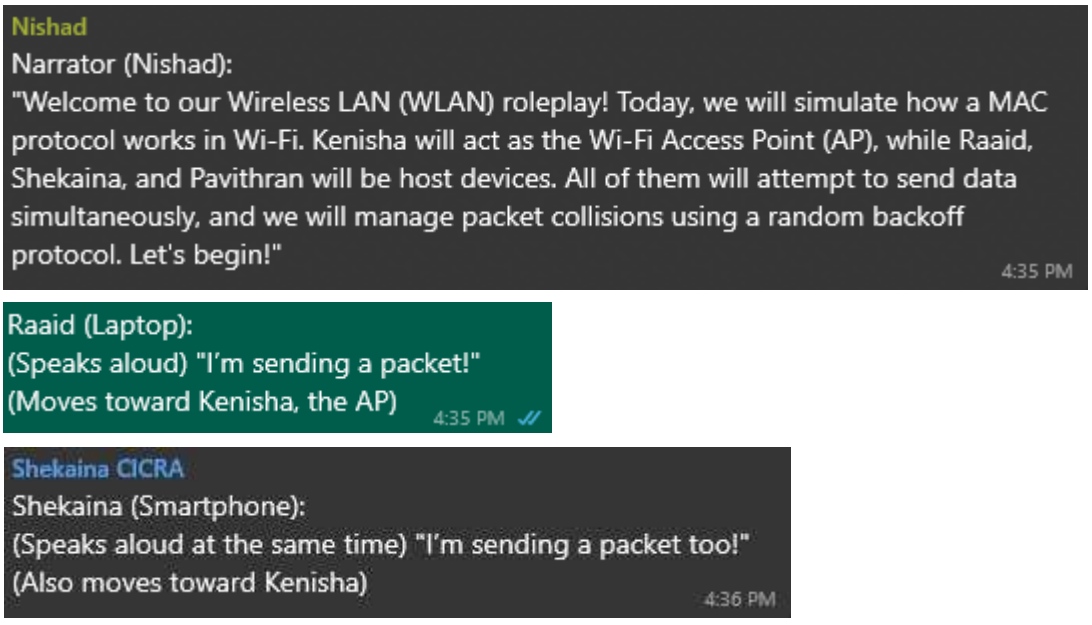
Members in this group activity task:

Nishad – C23110001
Kenisha – C23020001
Shekaina – C23110002
Raaid – C23020004
Pavithran – C22060015

Activity 1

1. Let's do a small role play to understand the MAC protocol. Assume that your group forms a Wireless LAN (WLAN) that uses Wi-Fi technology (has a wireless access point (AP) and three wireless devices that connect to Wi-Fi AP). One member can be the Wi-Fi AP and other members are the hosts (could be laptops, smart watches, smart phones, etc.). Assume all the devices in the network would like to send packets to Internet simultaneously. For example, when you need to send a packet, you can say "I'm sending a packet". If another group member said the same thing at the same time, then a collision occurred, and both need to retransmit packets.

   Your group needs to act as a Wireless LAN and provides a mechanism to enable successful packet transmission. You can illustrate the protocol that you have designed in a timing diagram (Shows in Figure 1). You are not required to replicate the exact protocol used in Wi-Fi. You can design your own protocol based on random access that allows hosts to communicate with Wi-Fi AP with minimal collisions and act fast in case of a packet collision.



Nishad
Nishad (Observer):
(Calls out) "Collision detected! Both Raaid and Shekaina sent packets at the same time."
4:37 PM

Kenisha CICRA
Kenisha (Wi-Fi AP):
(Speaks after detecating the collision) "Collision detected! Both Raaid and Shekaina need to retransmit."
4:40 PM

Nishad
Nishad (Observer):
(Sets a short timer for random backoff intervals) "Now, each host must wait a random amount of time before retransmitting their packets. Raaid will wait 2 seconds, and Shekaina will wait 3 seconds."
4:41 PM

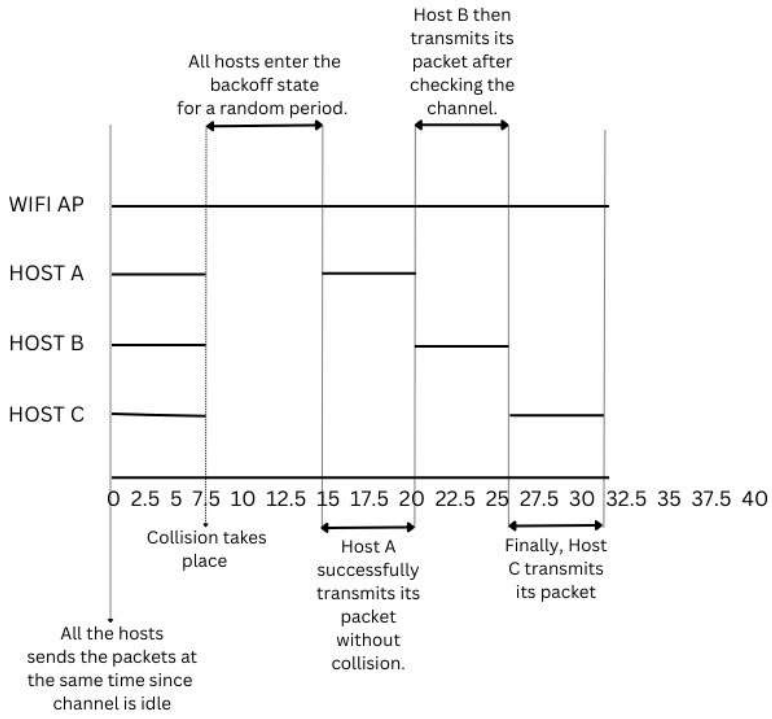
Raaid (Laptop):
(After 2 seconds, speaks aloud) "I'm sending a packet again after my backoff!"
(Moves toward Kenisha successfully this time)
4:42 PM

Kenisha CICRA
Kenisha (WiF-i AP):
(Responds) "Packet received successfully from Raaid"
6:31 PM

Shekaina CICRA
Shekaina (Smartphone):
(After 3 seconds, speaks aloud) "I'm sending my packet again after my backoff!"
(Moves toward Kenisha)
8:05 PM

Kenisha CICRA
Kenisha (Wi-Fi AP):
(Responds) "Pack received successfully from Shekaina!"
8:16 PM

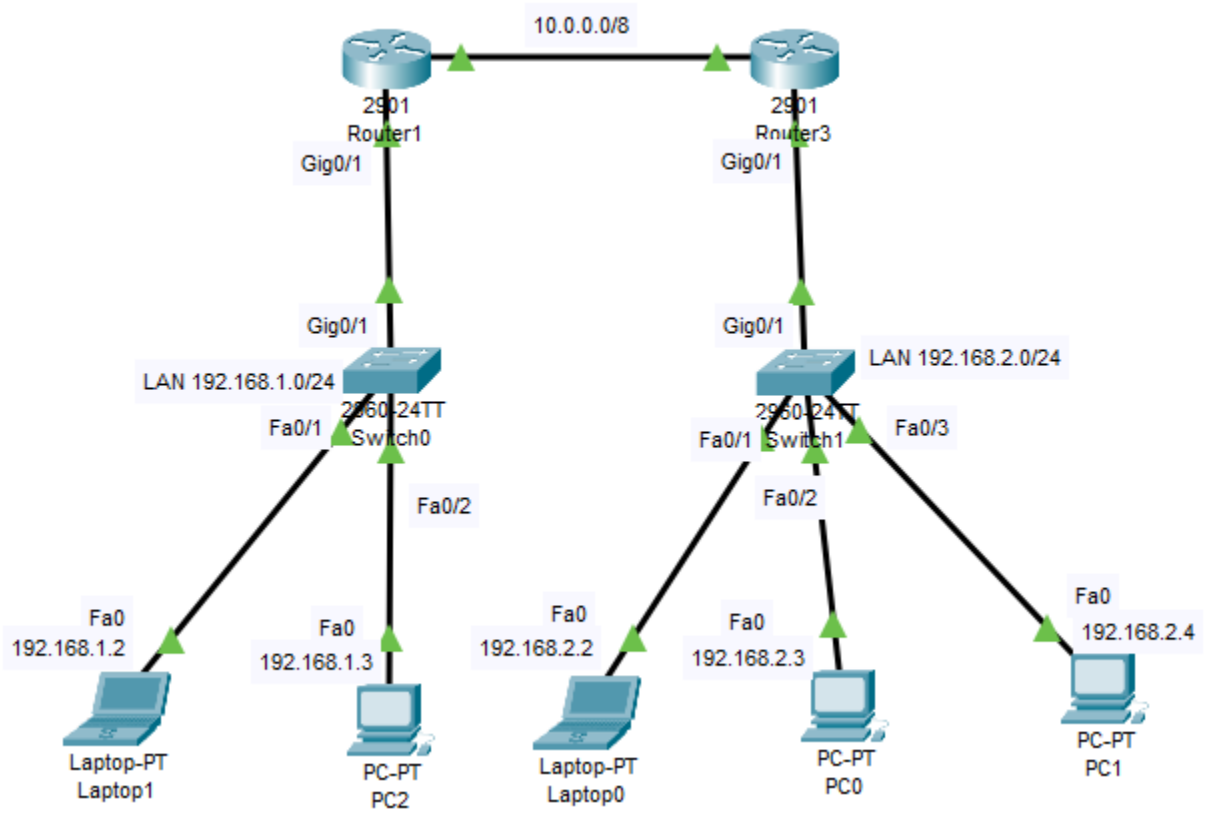
Nishad
Narrator (Nishad):
"As we've seen, our MAC protocol managed packet collisions using a random backoff strategy. By spacing out retransmissions after a collision, we minimized the chances of further collisions, allowing each device to send its packet successfully. This simple protocol illustrates how random access works to manage shared wireless medium access efficiently."
9:08 PM



2. Once you have completed the above activity, discuss the following question with your group members.
    - What is the medium access control (MAC) protocol that can be used in WiFi?

The MAC Address protocol used in Wi-Fi is Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). this protocol mechanism helps WIFI devices share the wireless medium efferently therefore reducing the chances of data collision and improving overall performance.

Activity 2



Implement the above network in Cisco Packet Tracer. You can use static IP configuration to configure hosts and router interfaces. Make sure to take screenshots of your findings as you need to include those in the task submissions.

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface GigabitEthernet0/1
Router(config-if)#ip address 192.168.1.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state to
up

Router(config-if)#exit
Router(config)#interface GigabitEthernet0/0
Router(config-if)#ip address 10.0.0.1 255.0.0.0
Router(config-if)#ip address 10.0.0.1 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0, changed state to
up

Router(config-if)#exit
Router(config)#
Router(config)#ip route 192.168.2.0 255.255.255.0 10.0.0.2
Router(config)#
```

Router1 Configuration

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface GigabitEthernet0/1
Router(config-if)#ip address 192.168.2.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state to
up

Router(config-if)#exit
Router(config)#interface GigabitEthernet0/0
Router(config-if)#ip address 10.0.0.2 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0, changed state to
up

Router(config-if)#exit
Router(config)#
Router(config)#ip route 192.168.1.0 255.255.255.0 10.0.0.1
Router(config)#
```

Router3 Configuration

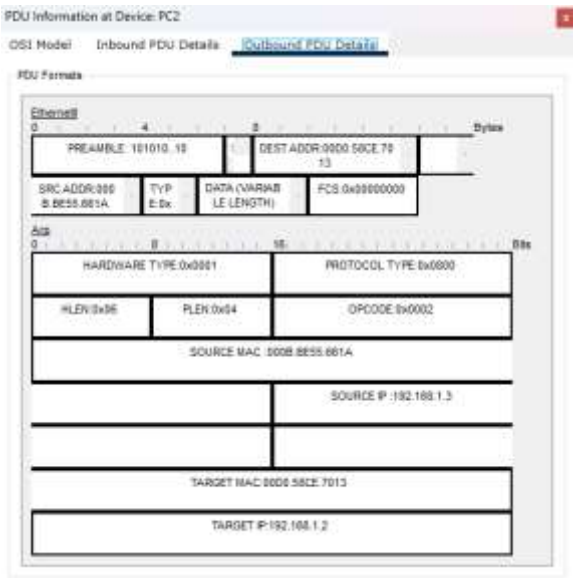
1. As a group, discuss what information Laptop1 in LAN1 requires to connect to PC2 In LAN1.

    - Laptop1 needs the IP address and MAC address of PC2 to communicate with it.

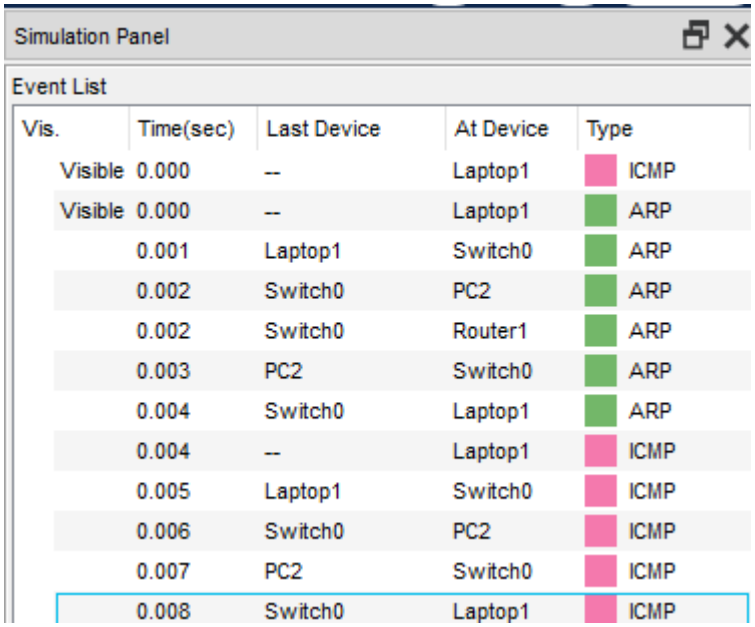
2. What protocol we can use to get the required information? Discuss the steps involved in getting the required information.

    - We can use Address Resolution Protocol (ARP) protocol where an end device sends a broadcast message asking for an IP address of another end device which is wants to communicate with, then the destination end device replies with its IP address and MAC address and the source end device saves it on its ARP table.

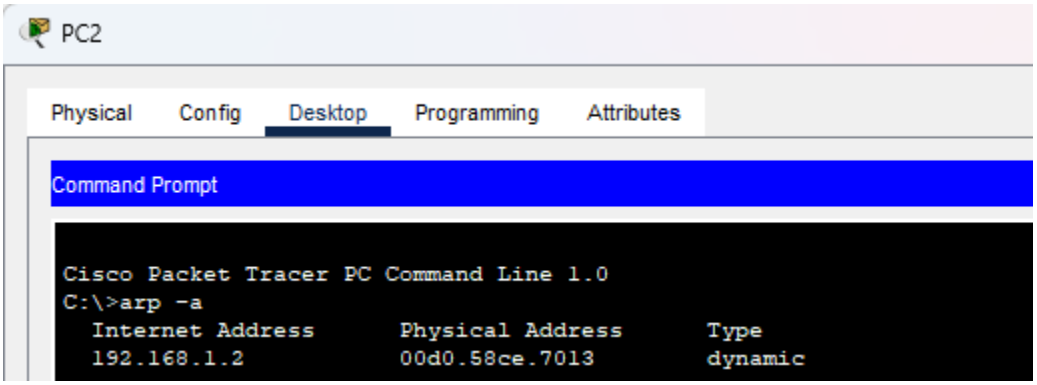
3. Use the simulation mode to check ARP in action by pinging PC2 from Laptop1.
    a. What are the types of messages that PC2 generated?



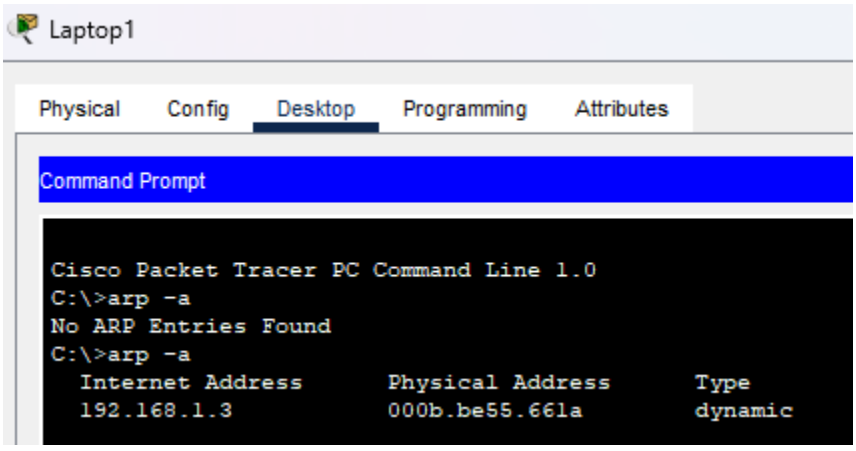
b. Discover the message sequence of ARP and the message content (paying attention to the source IP, destination IP, source MAC address, and destination MAC address).



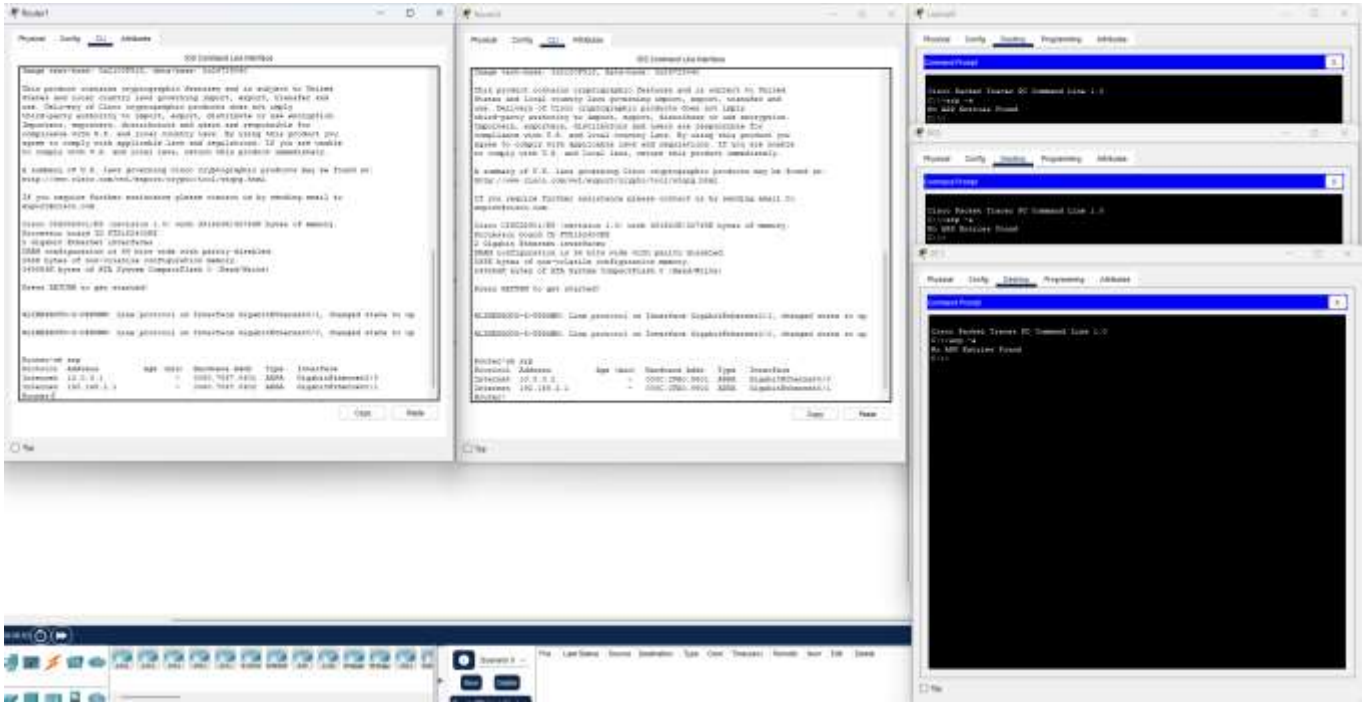
4. Each device maintains an ARP table. You can check the ARP table of devices using the command prompt and typing "arp -a". Check and compare the arp tables in Laptop 1 and PC2.



PC2's ARP table contains the IP and MAC address of Laptop1 and as seen below Laptop1's ARP table holds the IP and MAC addresses of PC2

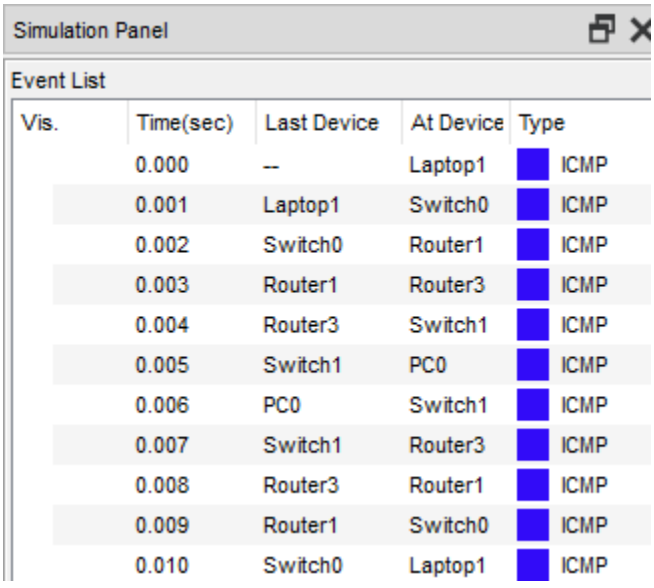


5. Check the arp tables of router 1, router 2, PC0, PC1, and Laptop0. Keep notes of the content of each arp table. To show the arp table of a router, “show arp” command can be used in the router’s CLI.



Currently both router only have the IP and MAC addresses of their connection to the switch and the other router and the end devices do not have anh entries in their ARP table.

6. Use the simulation mode again. Now, ping PC0 from Laptop1. Discover the arp message sequence and the message content in each link. Once the above steps are completed, ping PC1 from laptop0.



Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	Laptop1	ICMP
	0.001	Laptop1	Switch0	ICMP
	0.002	Switch0	Router1	ICMP
	0.003	Router1	Router3	ICMP
	0.004	Router3	Switch1	ICMP
	0.005	Switch1	PC0	ICMP
	0.006	PC0	Switch1	ICMP
	0.007	Switch1	Router3	ICMP
	0.008	Router3	Router1	ICMP
	0.009	Router1	Switch0	ICMP
	0.010	Switch0	Laptop1	ICMP

ICMP, sequence once the ARP request was replied between Laptop0 and PC0

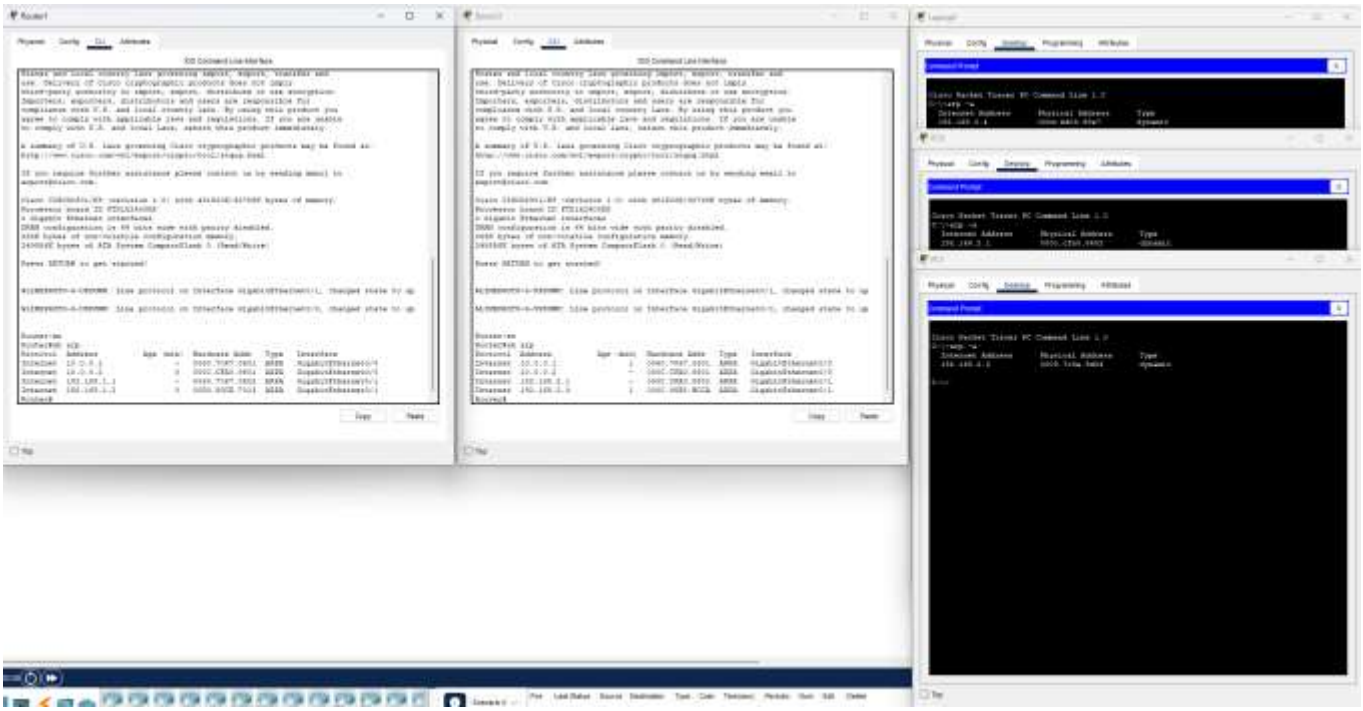
Simulation Panel

Event List

Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	Laptop0	ICMP
	0.000	--	Laptop0	ARP
	0.001	Laptop0	Switch1	ARP
	0.002	Switch1	PC0	ARP
	0.002	Switch1	PC1	ARP
	0.002	Switch1	Router3	ARP
	0.003	PC1	Switch1	ARP
	0.004	Switch1	Laptop0	ARP
	0.004	--	Laptop0	ICMP
	0.005	Laptop0	Switch1	ICMP
	0.006	Switch1	PC1	ICMP
	0.007	PC1	Switch1	ICMP
Visible	0.008	Switch1	Laptop0	ICMP

Successful ARP sequence leading to the aftermath ICMP sequence as well between Laptop0 and PC1

7. Check the arp tables of router 1, router 2, PC0, PC1, and Laptop0.



Updated ARP tables

8. Compare your observations with the observations you recorded in step 5. Discuss your finding with your group members.
    - We can see that the reason both routers and all end devices in the 192.168.2.0 subnet have updated their ARP tables is because by sending ARP request to send a ping any end device, be it in your own subnet or not if there is a reply, you will get the receivers IP and MAC address as you can see in the diagrams above.

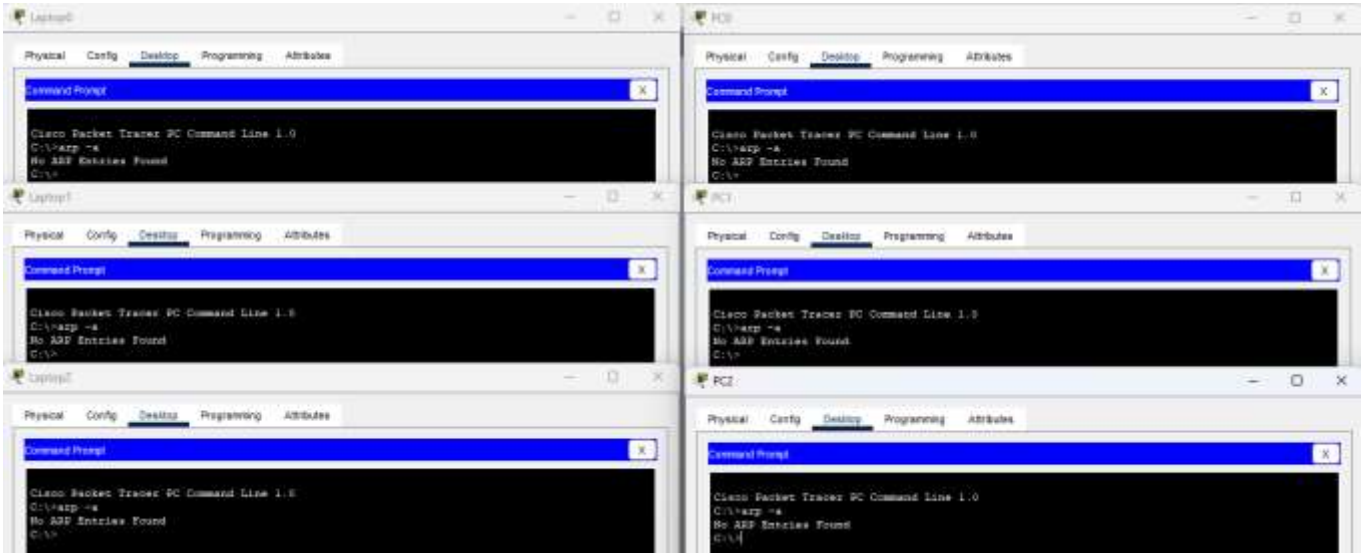
Activity 3

For this activity, you need to build and configure the following topology using a Cisco 2960 Switch.

1. After you configured the devices using static IP configuration,
    a. Record the MAC addresses of PCs/laptops/servers and Ethernet Ports of Switches.

```
Device Name: Switch0
Custom Device Model: 2960 IOS15
Hostname: Switch

Port                 Link   VLAN   IP Address          MAC Address
FastEthernet0/1      Up     1      --                  000C.CF34.2D01
FastEthernet0/2      Up     1      --                  000C.CF34.2D02
FastEthernet0/3      Up     1      --                  000C.CF34.2D03
FastEthernet0/4      Up     1      --                  000C.CF34.2D04
FastEthernet0/5      Up     1      --                  000C.CF34.2D05
FastEthernet0/6      Up     1      --                  000C.CF34.2D06
FastEthernet0/7      Up     1      --                  000C.CF34.2D07
FastEthernet0/8      Down   1      --                  000C.CF34.2D08
FastEthernet0/9      Down   1      --                  000C.CF34.2D09
FastEthernet0/10     Down   1      --                  000C.CF34.2D0A
FastEthernet0/11     Down   1      --                  000C.CF34.2D0B
FastEthernet0/12     Down   1      --                  000C.CF34.2D0C
FastEthernet0/13     Down   1      --                  000C.CF34.2D0D
FastEthernet0/14     Down   1      --                  000C.CF34.2D0E
FastEthernet0/15     Down   1      --                  000C.CF34.2D0F
FastEthernet0/16     Down   1      --                  000C.CF34.2D10
FastEthernet0/17     Down   1      --                  000C.CF34.2D11
FastEthernet0/18     Down   1      --                  000C.CF34.2D12
FastEthernet0/19     Down   1      --                  000C.CF34.2D13
FastEthernet0/20     Down   1      --                  000C.CF34.2D14
FastEthernet0/21     Down   1      --                  000C.CF34.2D15
FastEthernet0/22     Down   1      --                  000C.CF34.2D16
FastEthernet0/23     Down   1      --                  000C.CF34.2D17
FastEthernet0/24     Down   1      --                  000C.CF34.2D18
GigabitEthernet0/1   Up     1      --                  000C.CF34.2D19
GigabitEthernet0/2   Down   1      --                  000C.CF34.2D1A
Vlan1                Down   1      <not set>           00E0.F7C3.A9BD

Physical Location: Intercity > Home City > Corporate Office > Main Wiring Closet > Rack > Switch0
```

b. Record the arp table in PCs and laptops.



c. Check the MAC address table of the switch. In switch's CLI, you can type the following command to show the MAC address table.

```
Switch>en
Switch#sh mac-add
Switch#sh mac-address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----

   1    0060.3e69.9001    DYNAMIC     Gig0/1
Switch#
```

d. Record your observations. If there are any records in the MAC table, explain your observation.

- The switch currently only has the router's MAC Address since the router is connected with its gig ethernet port to the switch's gig ethernet port thus letting the switch to record the routers mac addresses with the port its connected to.

e. Now, ping from PC1 and PC2 to Laptop0 and Laptop 1, respectively.

Simulation Panel

Event List

Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	PC1	ICMP
	0.000	--	PC1	ARP
	0.001	PC1	Switch0	ARP
	0.002	Switch0	Server0	ARP
	0.002	Switch0	PC0	ARP
	0.002	Switch0	PC2	ARP
	0.002	Switch0	Laptop0	ARP
	0.002	Switch0	Laptop1	ARP
	0.002	Switch0	Laptop2	ARP
	0.002	Switch0	Router4	ARP
	0.003	Laptop0	Switch0	ARP
	0.004	Switch0	PC1	ARP
	0.004	--	PC1	ICMP
	0.005	PC1	Switch0	ICMP
	0.006	Switch0	Laptop0	ICMP
	0.007	Laptop0	Switch0	ICMP
Visible	0.008	Switch0	PC1	ICMP

Pinging from PC1 to Laptop0 successfully

Simulation Panel

Event List

Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	PC2	ICMP
	0.000	--	PC2	ARP
	0.001	PC2	Switch0	ARP
	0.002	Switch0	Server0	ARP
	0.002	Switch0	PC0	ARP
	0.002	Switch0	PC1	ARP
	0.002	Switch0	Laptop0	ARP
	0.002	Switch0	Laptop1	ARP
	0.002	Switch0	Laptop2	ARP
	0.002	Switch0	Router4	ARP
	0.003	Laptop1	Switch0	ARP
	0.004	Switch0	PC2	ARP
	0.004	--	PC2	ICMP
	0.005	PC2	Switch0	ICMP
	0.006	Switch0	Laptop1	ICMP
	0.007	Laptop1	Switch0	ICMP
Visible	0.008	Switch0	PC2	ICMP

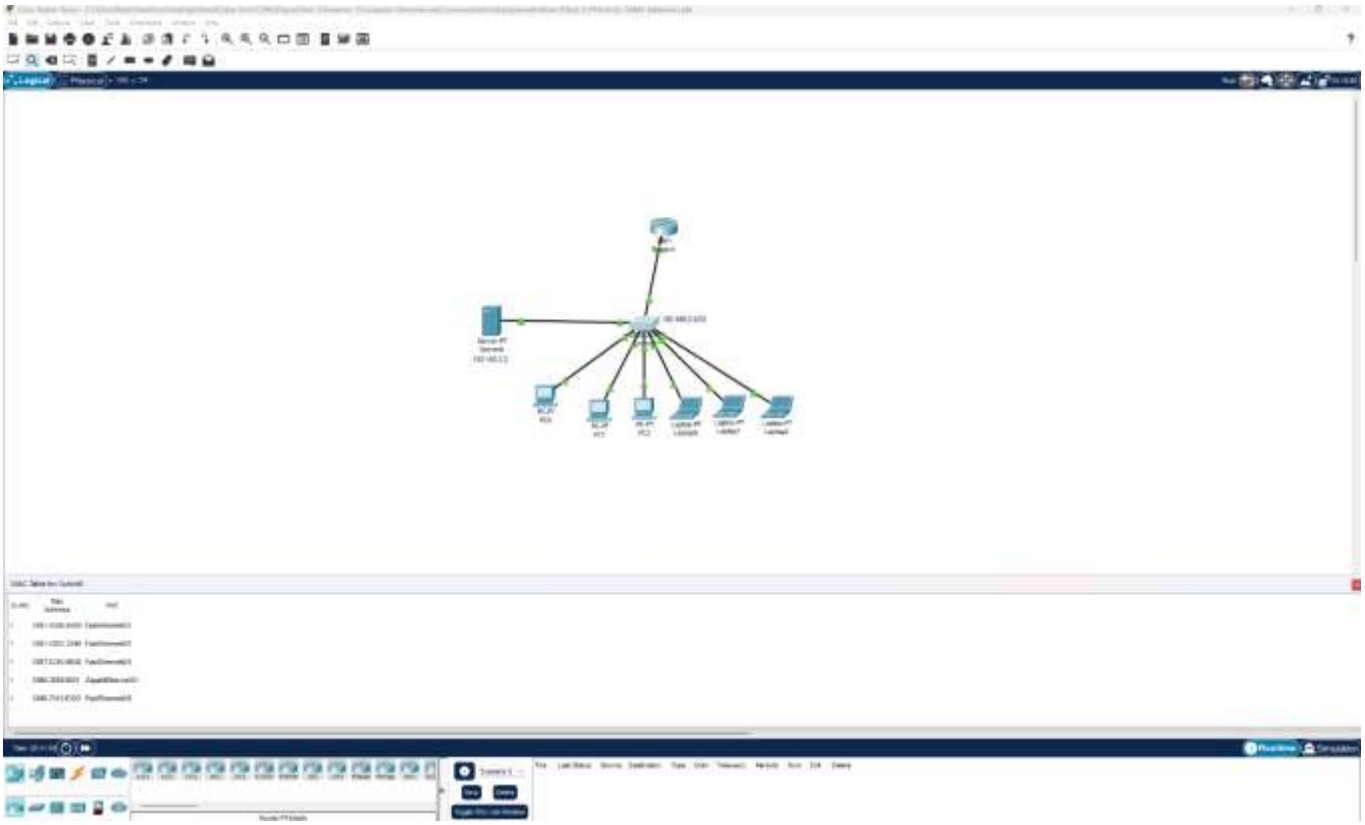
Pinging from PC2 to Laptop1 successfully

f. Check the MAC address table of the switch. Explain your observations.

```
Switch#sh mac-address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----

   1    0001.4389.ad68    DYNAMIC     Fa0/3
   1    0001.c92c.2046    DYNAMIC     Fa0/5
   1    0007.ec93.664d    DYNAMIC     Fa0/4
   1    0060.3e69.9001    DYNAMIC     Gig0/1
   1    0060.7013.e30d    DYNAMIC     Fa0/6
```

After the ARP request and reply. The MAC Address table has been enlarged with the details of ARP packets sent by end devices being recorded as they arrived at the switch.

g. Click on the "magnifying glass" icon and bring that on top of the switch. Click on the switch and select "MAC table". Resize the MAC address table and keep the table visible.

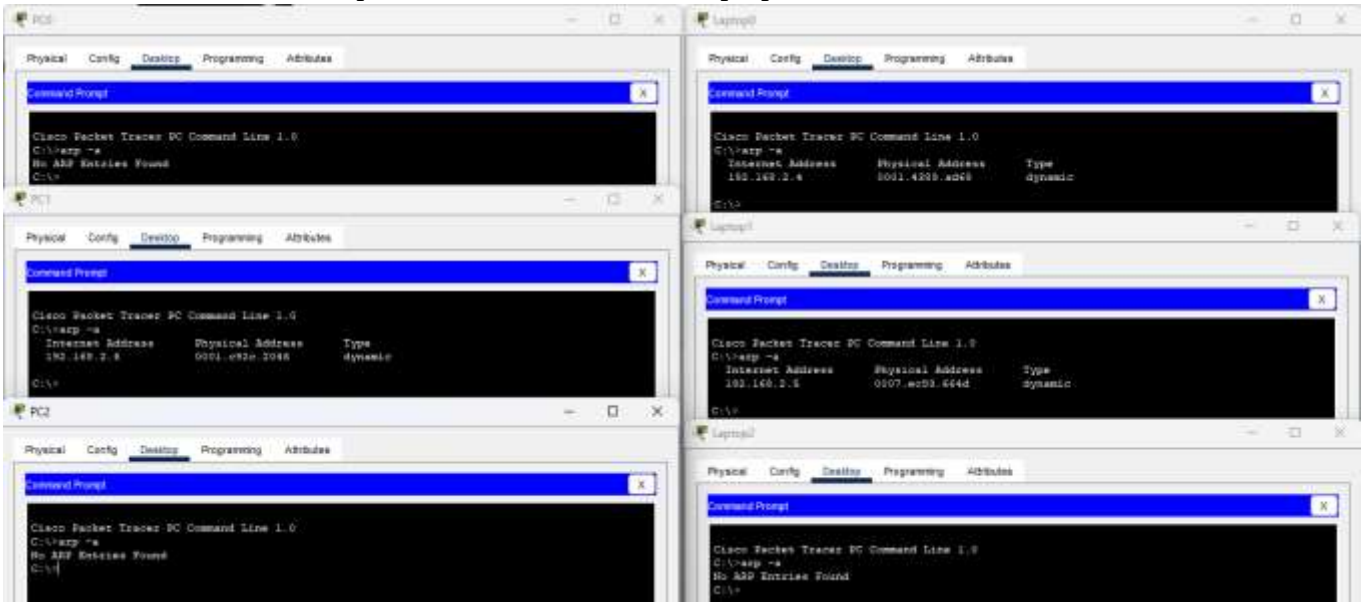


h. Ping laptop2 from PC0 and check the changes in the MAC table. Explain your observation.

```
1        00E0.B0C4.3B47  FastEthernet0/7
```

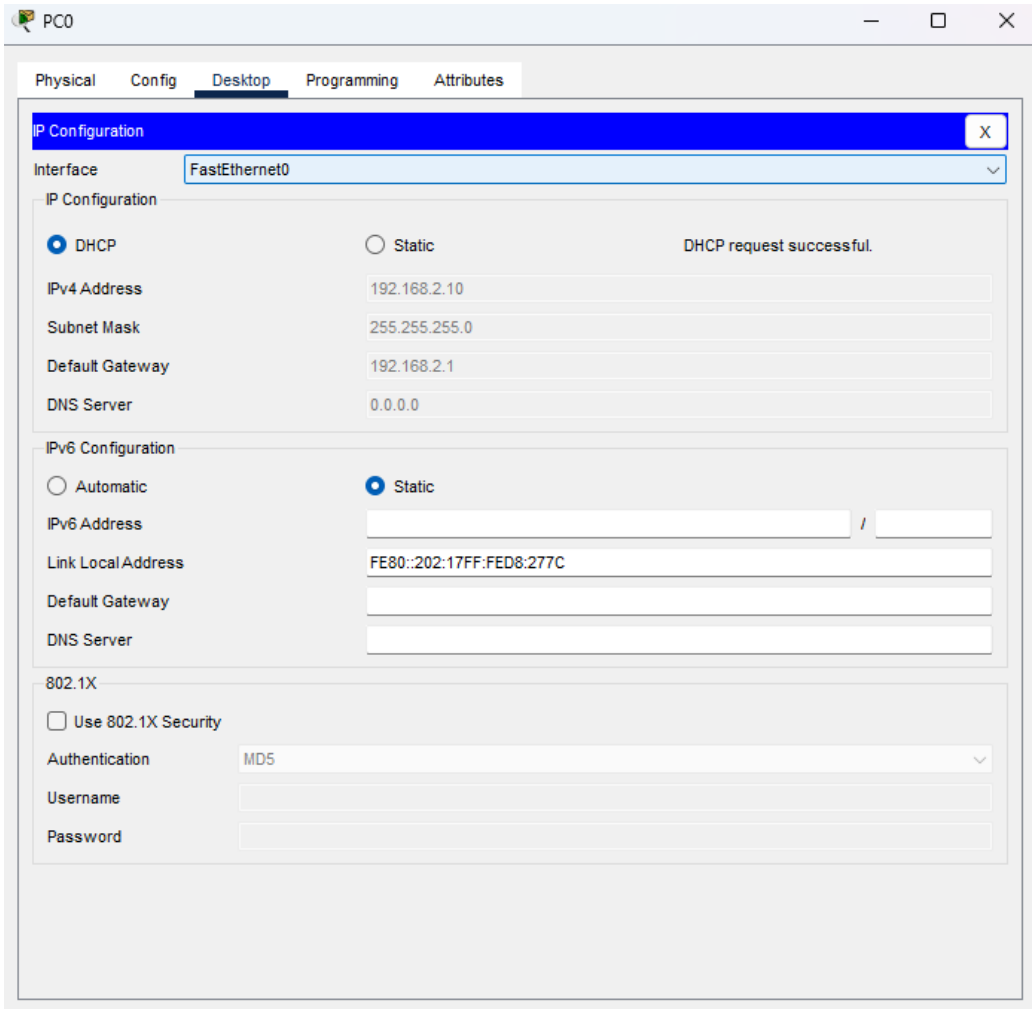
- A new MAC Address entry has been added after the Pinging Laptop2 from PC0. Since they hadn't communicated before, PC0 sent out an ARP broadcast request and Laptop2 replied by sending its MAC Address and the switch recorded it

    i. Check the arp tables in all the PCs and laptops.



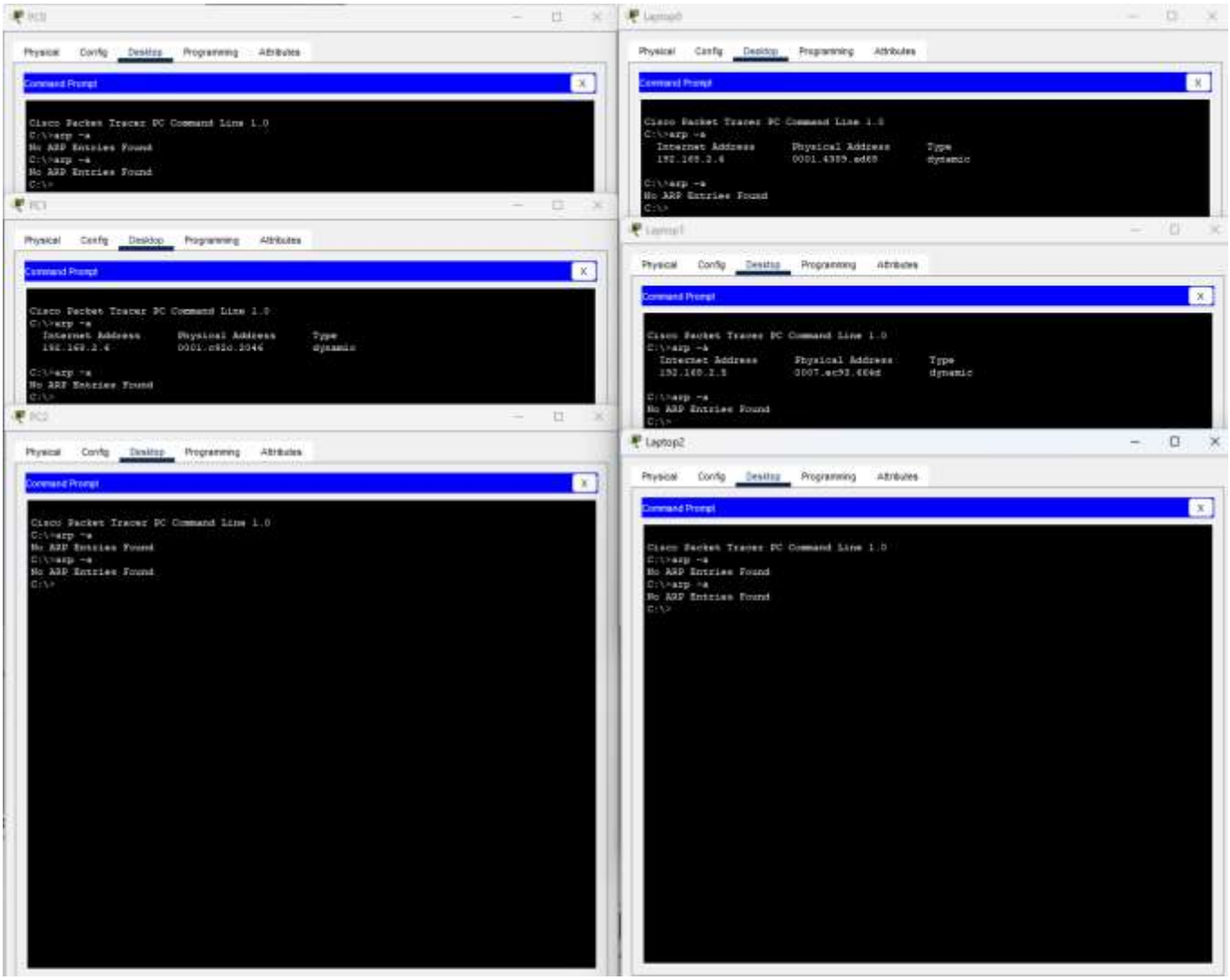
2. Clear the mac address table from the switch. You can do this by using "clear mac- address-table" command in CLI of the switch.
    a. Configure server as a DHCP server and use DHCP to obtain IP address for all PCs and laptops.

Configuring server by adding a DHCP pool



Setting IP Configuration in end devices to DHCP as seen in PC0

b. Check the Arp table of PCs and laptops. Compare you observation with what you have recorded in 1.b).



c. Check the Mac table of the switch. Compare and explain you observation with what you have recorded in 1.c)

```
Switch>en
Switch#sh mac-a
Switch#sh mac-address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----

   1    0001.4389.ad68    DYNAMIC     Fa0/3
   1    0001.c92c.2046    DYNAMIC     Fa0/5
   1    0002.17d8.277c    DYNAMIC     Fa0/2
   1    0006.2a89.237e    DYNAMIC     Fa0/1
   1    0007.ec93.664d    DYNAMIC     Fa0/4
   1    0060.3e69.9001    DYNAMIC     Gig0/1
   1    0060.7013.e30d    DYNAMIC     Fa0/6
   1    00e0.b0c4.3b47    DYNAMIC     Fa0/7
```

The switch automatically records the ARP packets it receives by the end devices who are sending a broadcast message to the server to receive a address from the DHCP server pool.

Above and Beyond Tasks

1. Discuss the security issues associated with ARP identifying the types of attacks (200 – 300 words).

ARP's absence of authentication procedures leaves it open to many security concerns, despite its crucial role in mapping IP addresses to MAC addresses within a network. Now, let's talk about its shortcomings.

ARP Spoofing

ARP spoofing is one of the most popular attacks that take use of ARP vulnerabilities. In order to carry out this attack, a malevolent actor would forge ARP messages and send them into the network, linking their MAC address to the IP address of a victim host or gateway. Consequently, data meant for the authorized device may be intercepted, altered, or blocked by the attacker. This could serve as the basis for a number of attacks, including

1. Man-in-the-Middle – ARP spoofing is used by the attacker in an MITM attack to place himself in the middle of two communicating devices. This gives them the ability to intercept or modify the data being delivered, which can result in data manipulation, information theft, or session hijacking.
2. Denial of Service (DoS) – By associating fictitious MAC addresses with IP addresses through ARP spoofing, an attacker can essentially render the target device unavailable, causing interruptions to the network and a denial of service.
3. Session Hijacking – After employing ARP spoofing to intercept traffic, an attacker can hijack ongoing sessions between two authentic hosts by inserting malicious commands or obtaining private session tokens, like login passwords.

SIT202: Computer Networks and Communication
Leaning Evidence for Active Class Task 10

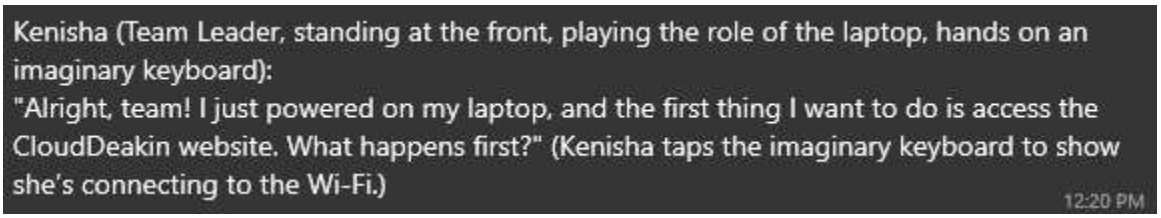
Name: Kenisha Corera
Student ID: C23020001

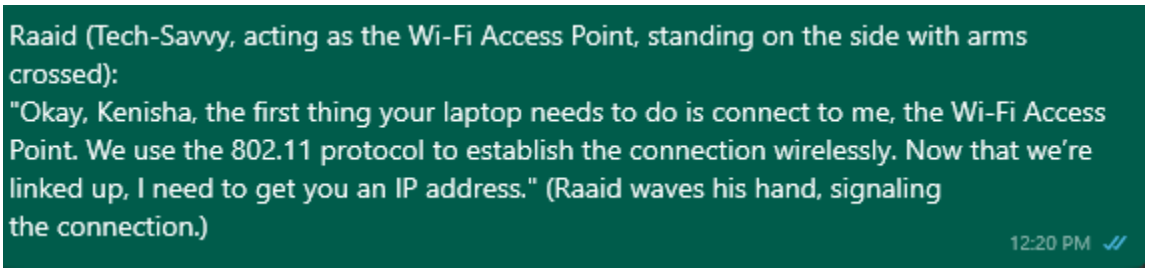
Members in this group activity task:

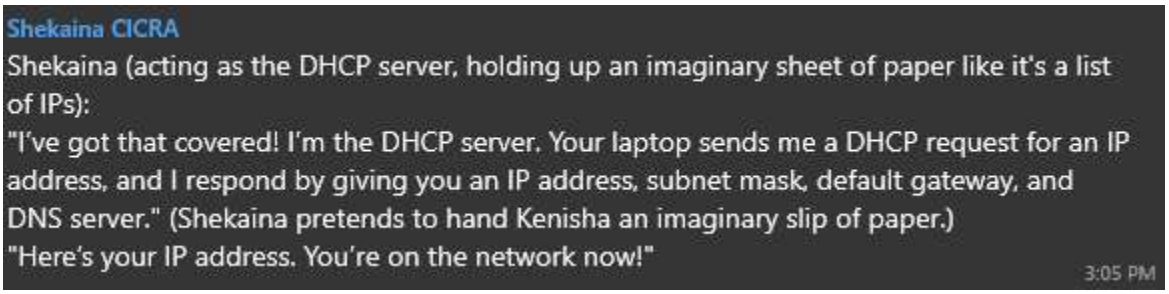
Nishad – C23110001
Kenisha – C23020001
Shekaina – C23110002
Raaid – C23020004
Pavithran – C22060015

Activity 1

1. Outline the major steps used by your laptop after it is first powered on until it downloads the page from CloudDeakin.
2. For each of the major steps you have outlined, identify the network protocols that are used and explain what functionality they provide in achieving the task.
3. Explain what would change in your answer to the above questions if your home network uses NAT.







Kenisha CICRA
Kenisha (still playing the laptop, holding the "IP address"):
"Thanks! Now I need to connect to the CloudDeakin website. I'll need the IP address of the server for 'd2l.deakin.edu.au'. Who's got that?" (Kenisha looks around.)
3:49 PM

Pavithran AIR
Pavithran (acting as the DNS server, waving like he's waiting for his turn):
"Over here! I'm the DNS server. You ask me for the IP address of 'd2l.deakin.edu.au'." (Pretending to search a database, Pavithran then points at Kenisha.)
"I found it! The IP address is 192.168.1.100. Now you can connect!"
3:55 PM

Kenisha CICRA
Kenisha (receiving the "IP address" from Pavithran):
"Awesome, got it! Now I need to establish a connection with the CloudDeakin server. Time for TCP to come into play. Nishad, I'm sending a SYN packet to you!" (Kenisha mimics sending a data packet.)
4:14 PM

Nishad
Nishad (acting as the CloudDeakin server, giving a nod and holding out his hand to catch the 'SYN' packet):
"I got your SYN! Here's my SYN-ACK response!" (Nishad tosses an imaginary packet back to Kenisha.)
5:24 PM

Kenisha CICRA
Kenisha (pretending to catch the packet, and quickly responding):
"And here's my ACK! Now we have a reliable connection, Nishad." (Kenisha does a small celebratory move.)
"Now, Nishad, I'm requesting the web page from you using HTTPS."
5:32 PM

Nishad
Nishad (as the server, crossing his arms, standing confidently):
"I'm on it! Since you're using HTTPS, I'll make sure everything's secure and encrypted." (Pretending to hold up a secured lock.)
"Here's the web page data—encrypted, of course." (Tosses the 'data packets' back to Kenisha.)
5:33 PM

Kenisha CICRA
Kenisha (pretending to receive the data packets and act like the page loads on her "laptop" screen):
"And boom! The SIT202 CloudDeakin page is loaded. Looks good, right?" (Gestures to the others as if showing the web page on a laptop.)
5:33 PM

Raaid (raising a hand, excited):
"Hold on, Kenisha. What if your home network was using NAT? What would change?" (Smirking, challenging the situation.)
3:03 AM ✓✓

Pavithran CICRA

Pavithran (stepping forward, acting as the home router with NAT):
"Good question! If I, the home router, am using NAT, then instead of your private IP address being sent to Nishad, I'd replace it with my own public IP address before forwarding the request." (He points at Nishad, indicating how he's hiding Kenisha's real IP address.)
"When Nishad replies, I swap my public IP back for your private IP and pass it back to you. This way, all devices in the home network share my single public IP address."

8:07 PM

Shekaina CICRA

Shekaina (nodding along, stepping up to Kenisha, pretending to look over the "laptop screen"):
"So, Nishad only sees the public IP address, not Kenisha's actual private one. And the NAT does all this behind the scenes. Pretty cool!"

8:31 PM

Nishad CICRA

Nishad (crossing his arms, confidently):
"Yeah, and from my perspective as the server, I have no idea how many devices are using that public IP. It's like talking to one person, but there are actually many behind the scenes."

8:56 PM

Kenisha CICRA

Kenisha (nodding, arms crossed, addressing everyone):
"Exactly! NAT allows us to have multiple devices on a network, all using one public IP address. It's an efficient way of managing connections. So, with NAT or without, we can still access the CloudDeakin page. Mission accomplished, team!" (She smiles at the group.)

8:56 PM

Pavithran CICRA

Pavithran (with a smile, giving a thumbs up):
"Great teamwork! We just walked through the entire process of accessing a web page, covering all the major protocols—802.11, DHCP, DNS, TCP, HTTPS, and NAT."

8:56 PM

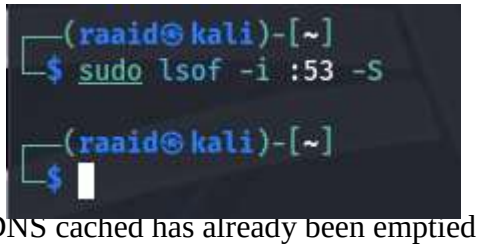
Kenisha CICRA

Kenisha (wrapping up, hands on hips):
"Fantastic job, everyone! I think we've got this networking thing down!" (Everyone cheers or gives high-fives.)

8:56 PM

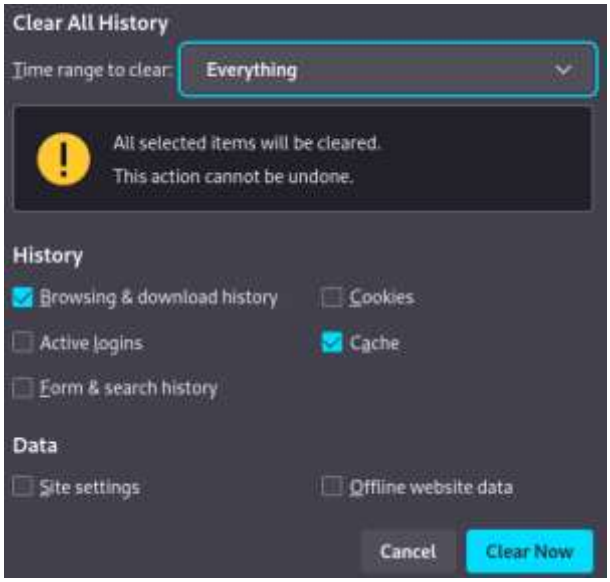
Activity 2

1. Use ipconfig in your command prompt/ terminal to empty the DNS cache in your host using ipconfig /flushdns. (MacOS Mojave use: sudo killall -HUP mDNSResponder)



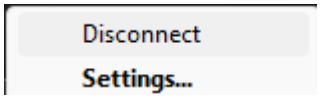
DNS cached has already been emptied

2. Open Google Chrome and empty your browser cache



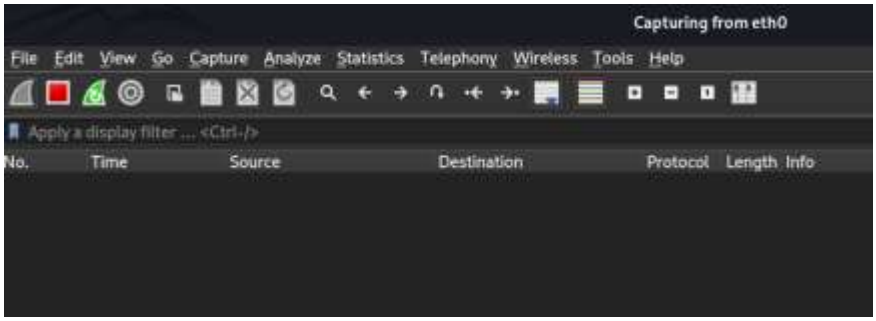
Browser cache has been cleared

3. Turn off your network connection (turn your WiFi off or disconnect your ethernet cable)

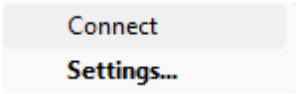


Switching off network connection in my Virtual Machine

4. Open Wireshark and start packet capture (remember to select your typical network interface). You will see an empty window.

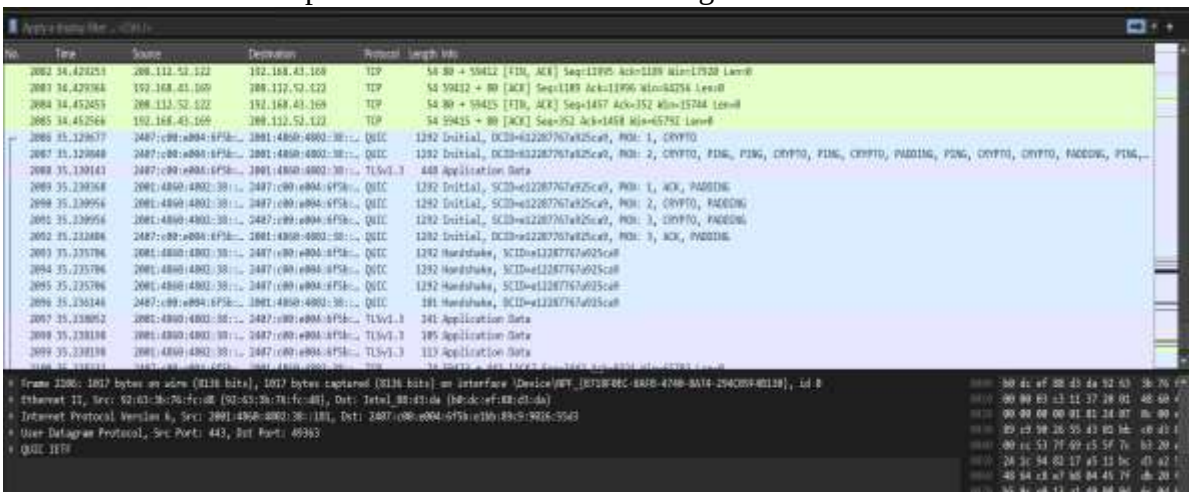


5. Turn on your network connection again.



6. Open your browser, visit the web page: http://www.discoverourtown.com and stop packet capture.

7. Now, you can start analyzing the packet capture.
    a) List down the order of protocols used and their messages.



1. QUIC (Quick UDP Internet Connections)

    - QUIC Client Hello: Client Hello, which is a message containing the client cipher and supported protocol versions by the client to the server is created to ensure a secure connection.
    - QUIC ACK: In order to confirm acknowledgement regarding received packets, both the client and server send ACK. Since it runs on UDP, it has its mechanism of ensuring data delivery in the connection to avoid instances where data is delivered unsuccessfully.
    - QUIC Data Packets: After the handshake, packets in the QUIC protocol are employed by both the client and the server to transfer encrypted data items such as requests and response to webpage.

2. TCP (Transmission Control Protocol)

    - SYN (Synchronize): The client initiates a connection by sending the server a message called SYN for connection.
    - SYN-ACK (Synchronize-Acknowledge): The request is accepted by the server; in reply the server sends a packet with SYN and ACK flags set.
    - ACK (Acknowledge): When the three way handshakes are over and the TCP connection has been established the client sends an ACK message back.
    - TCP Data Packets: Data residing at the client and server side are transmitted in the form of TCP segments after the connection has been made using the HTTP protocol.

3. TLSv1.3 (Transport Layer Security)

    - TLS Client Hello: Once the QUIC or TCP connection is successfully established then the client raises its hand by sending a Client Hello message that contains encryption preference and other session related information.
    - TLS Handshake: Both the client and the server share the encryption keys and decide upon secure sessions parameters.
    - TLS Encrypted Data: Once the handshake is done, any data is exchanged over the TCP or QUIC connection are encrypted by TLS to ensure the security of the data as it transmitted.

b) Explain the use of each protocol indicating the layer that they belong to.

1. QUIC (Quick UDP Internet Connections):

    - Purpose: QUIC may help clients and servers convey quicker and more safely. It ensures a minimal level of delays are acceptable while ensuring data encryption for security is upheld.
    - Layer: Unlike other protocols it adopts the UDP as the lower transport layer but works in the Fourth layer of the OSI model like TCP.

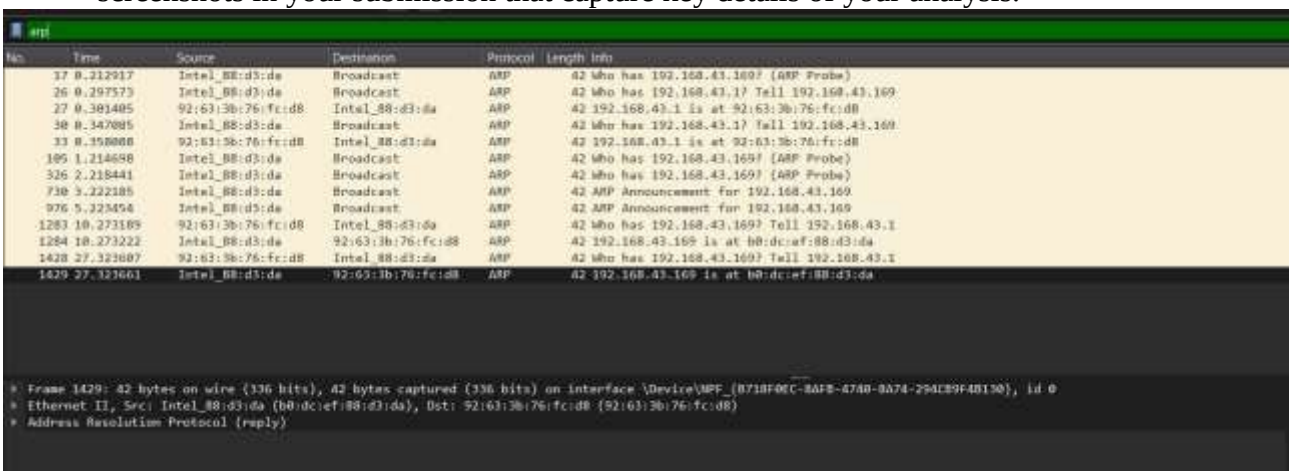
2. TCP (Transmission Control Protocol):

    - Purpose: By featuring control of data flow, error checking and retransmissions, TCP guarantees accurate message transmission by providing devices with reliable and timely information transfer.
    - Layer: WANs provide reliable data transmission at layer four also known as the Transport layer.

3. TLSv1.3 (Transport Layer Security):

    - Purpose: TLS encrypts data flow between clients and servers to ensure direct and indirect communications are secure and private.
    - Layer: HTTP, when used with TCP or QUIC, TLS runs at The Application Layer (Layer 7).

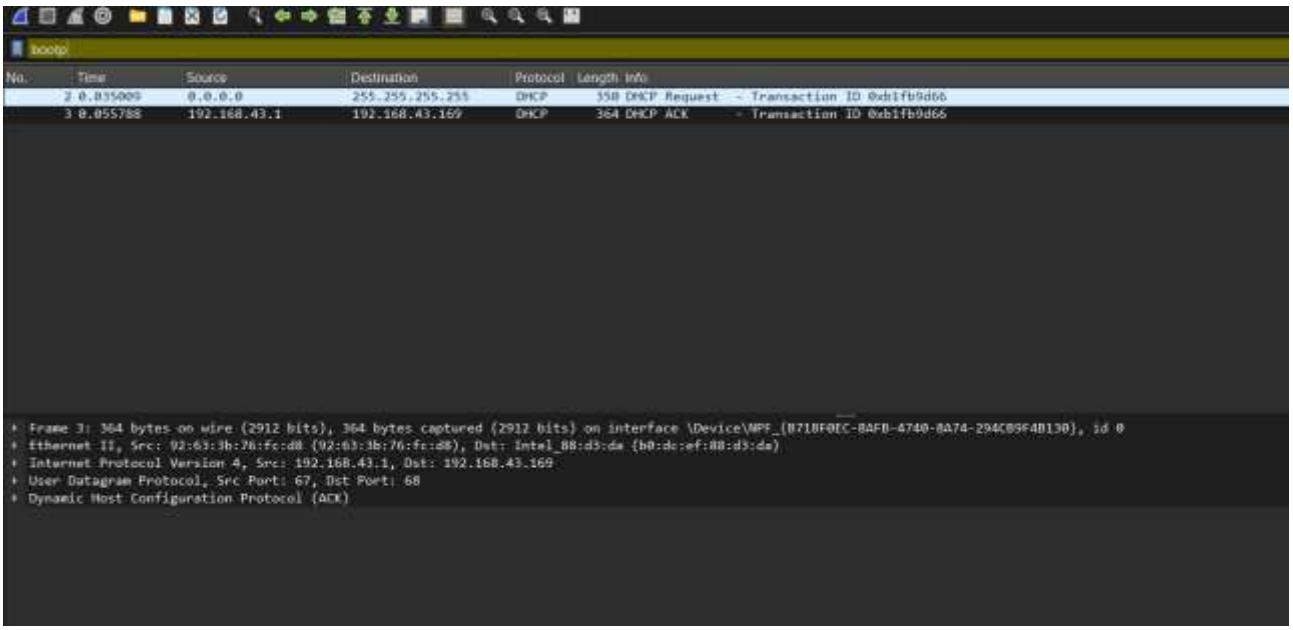
c) Check the details of each packet such as ARP, DHCP, DNS, and HTTP. Make sure you include screenshots in your submission that capture key details of your analysis.



The packet is an ARP Probe (or ARP Request) where the device with MAC address Intel_88:d3:da is asking the local network which device has the IP address 192.168.43.169.

- Address Resolution: Telecommunicating with an IP address of 192.168.43.169 means converting the IP address ad MAC address in the devices.
- Broadcast Request: This request is transmitted to all devices on the sending device network since the Address Resolution Protocol is unknown to the MAC address.

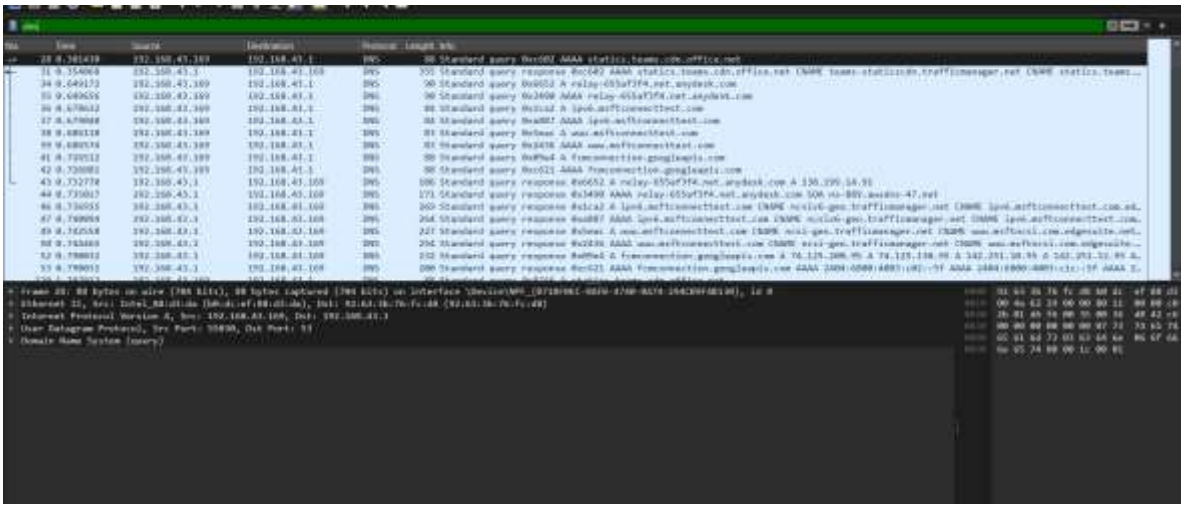
The device that controls IP address 192.168.43.169 will respond to this ARP Probe with MAC address. This is a network level query in a way to search for the MAC address equivalent to the given IP address.



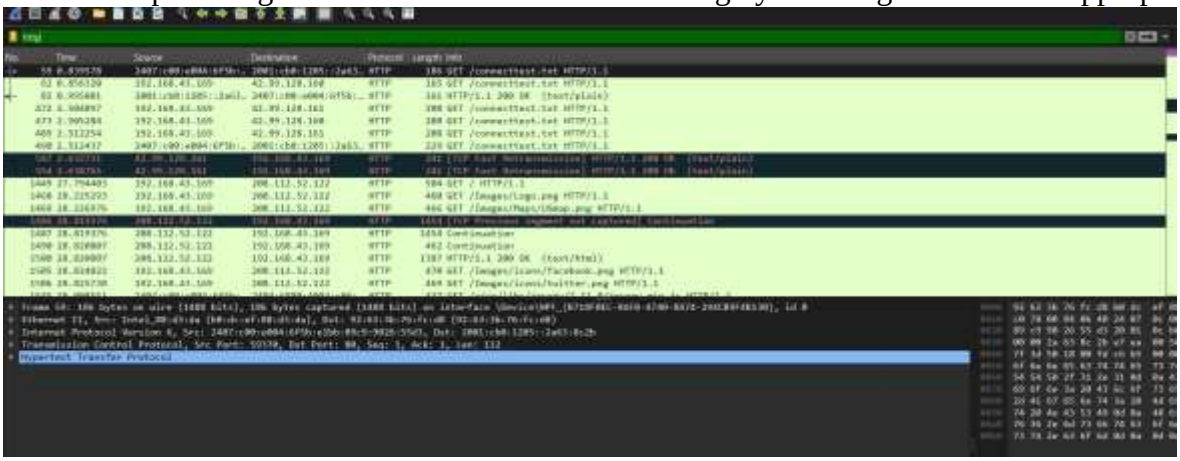
- This DHCP packet shows the result of a DHCP Request message that a client sends to ask for an IP address.

- Client Initialization: Right now, the client is attempting to get an IP address by connecting to a DHCP server mostly. The client does not possess an IP address yet, and thus the client broadcast for an IP address using the broadcast address of 255.255.255.255 so as to reach any and all DHCP servers.
- Transaction ID: 0xb1fb9d66 is used to keep track of the particular request in relation to the server's matching DHCP Offer or ACK.

This packet is a broadcast DHCP Request indicating that a client is trying to obtain an IP address configuration from any available DHCP server on the local network.



The DNS response is effectively redirecting requests from activity.windows.com to activity-consumer.trafficmanager.net and providing authoritative information about the DNS zone. This redirection helps manage domain names and load balancing by directing traffic to the appropriate servers.



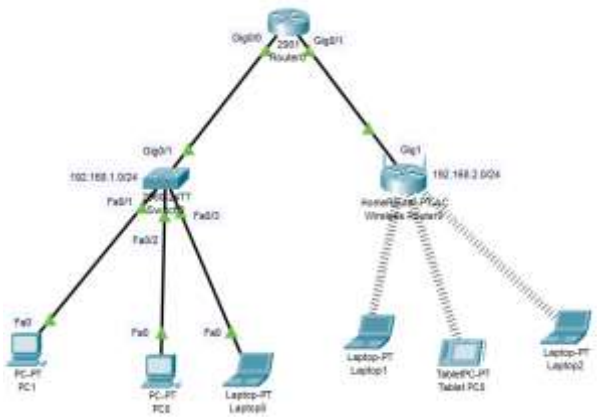
The packet with [TCP Fast Retransmission] indicates that a previously sent HTTP response (with a 200 OK status) was retransmitted due to a network issue or packet loss. The content of this packet is text/plain, which means it's a simple text file or response.

8. Compare the results of your analysis in the above step (7) with the findings of Activity 1.

- In the present activity, readers come to understand the function of networking protocols in the communication process as offered by activity 1 and 2. Activity 1 involves an interactive simulation of server, QUIC, TCP and TLS along with other components including Laptops, Wi-Fi Access Points, and NAT. The second activity is an extensive Packet Capture Analysis which is based purely on TCP, QUIC and TLS and displays how these protocols are relevant in the specific cases of website browsing.
- In brief, Activity 1 gives a general exposure of the networking paradigm, as in Activity 2, more details are added with regards to the packet level; therefore, both activities help enrich the knowledge on networking paradigms and their functionalities. Understanding of how these protocols work is facilitated by Activity 1, and Activity 2 provides a technical perspective of their objectives and levels of operation.

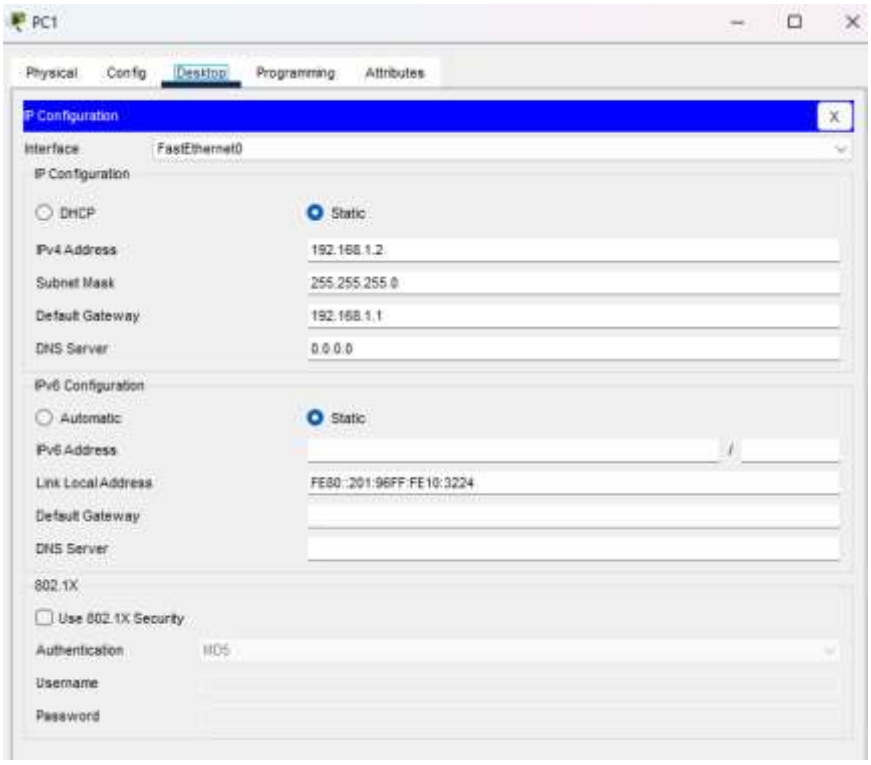
Activity 3

In this activity, you explore more on wired and wireless LANs. Implement the above network in Cisco Packet Tracer. You need to use static IP configuration to configure hosts and router interfaces. Make sure to take screenshots of your findings as you need to include the evidence in the task submissions

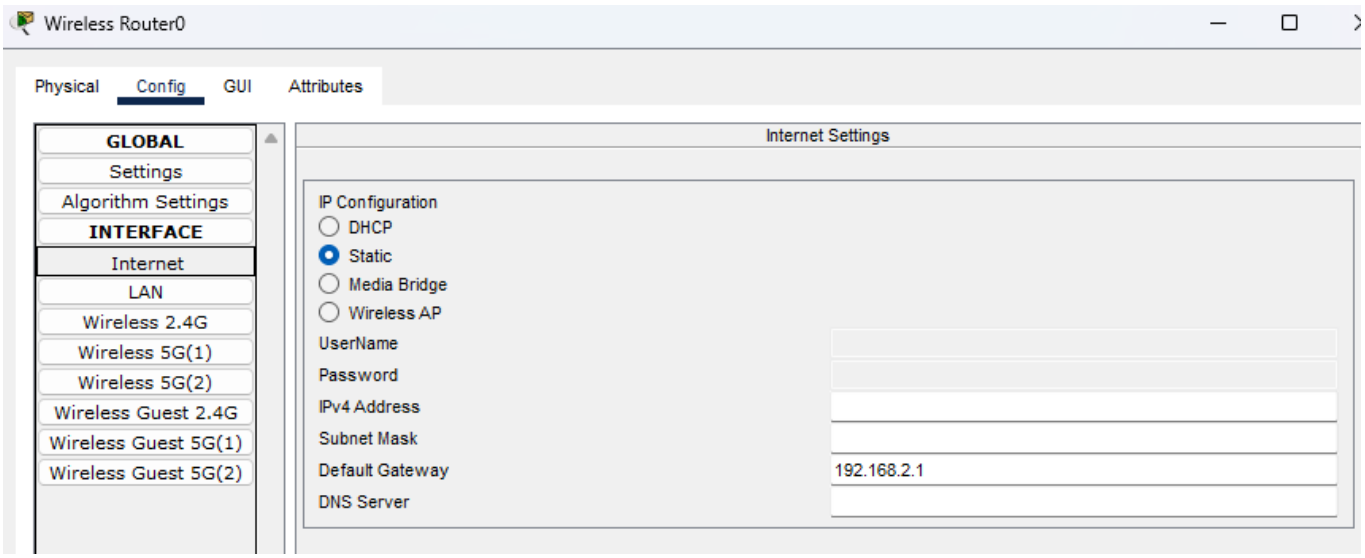


Implemented Network Architecture

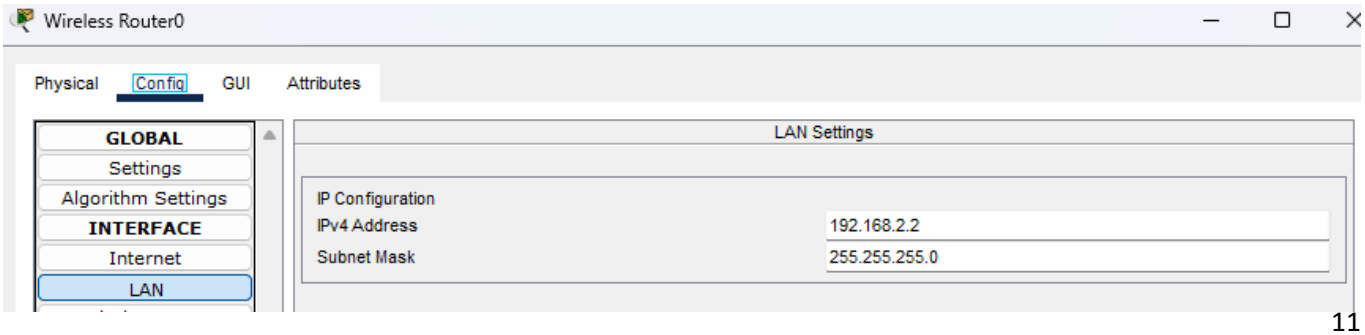
```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface GigabitEthernet0/0
Router(config-if)#ip address 192.168.1.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0, changed state to up

Router(config-if)#exit
Router(config)#interface GigabitEthernet0/1
Router(config-if)#ip address 192.168.2.1 255.255.255.0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state to up

Router(config-if)#exit
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
wr
Building configuration...
[OK]
Router#exit
```

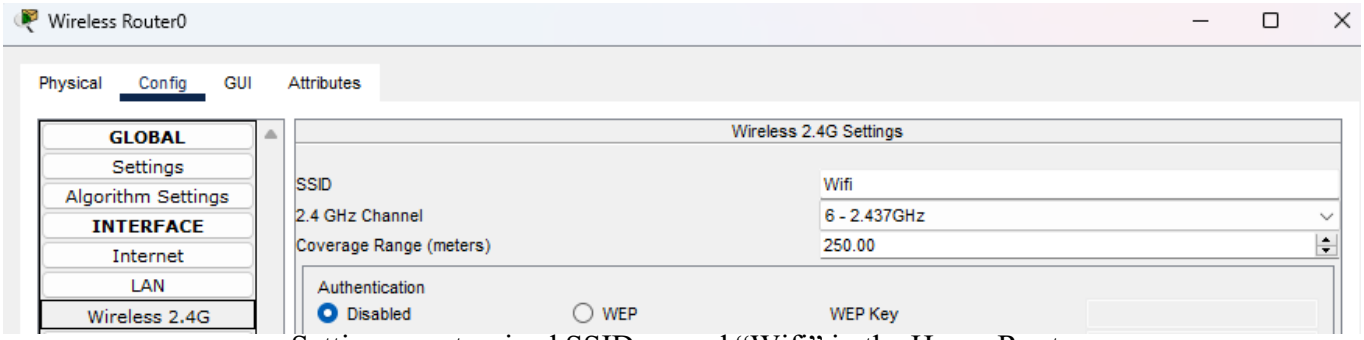


Setting up static connections with all end devices in both subnets as seen now in PC1

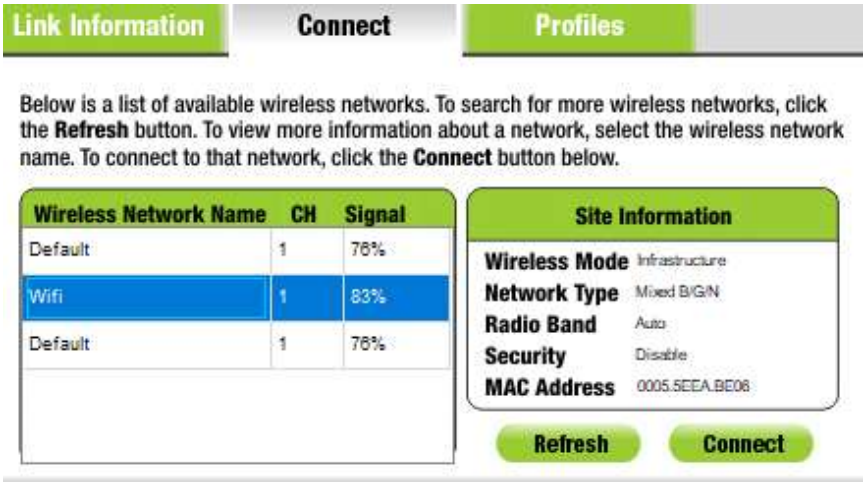


Configuring the Default Gateway for the Home Router



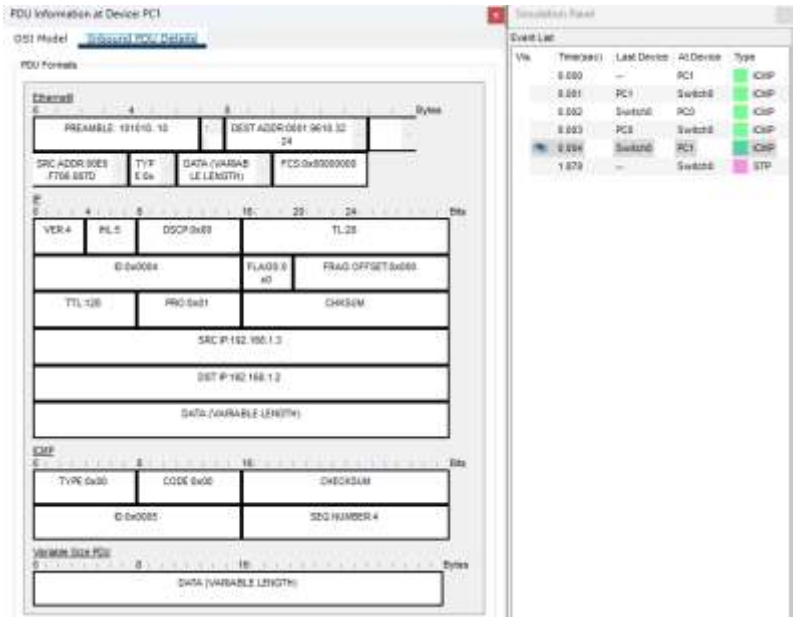


Setting a customized SSID named “Wifi” in the Home Router

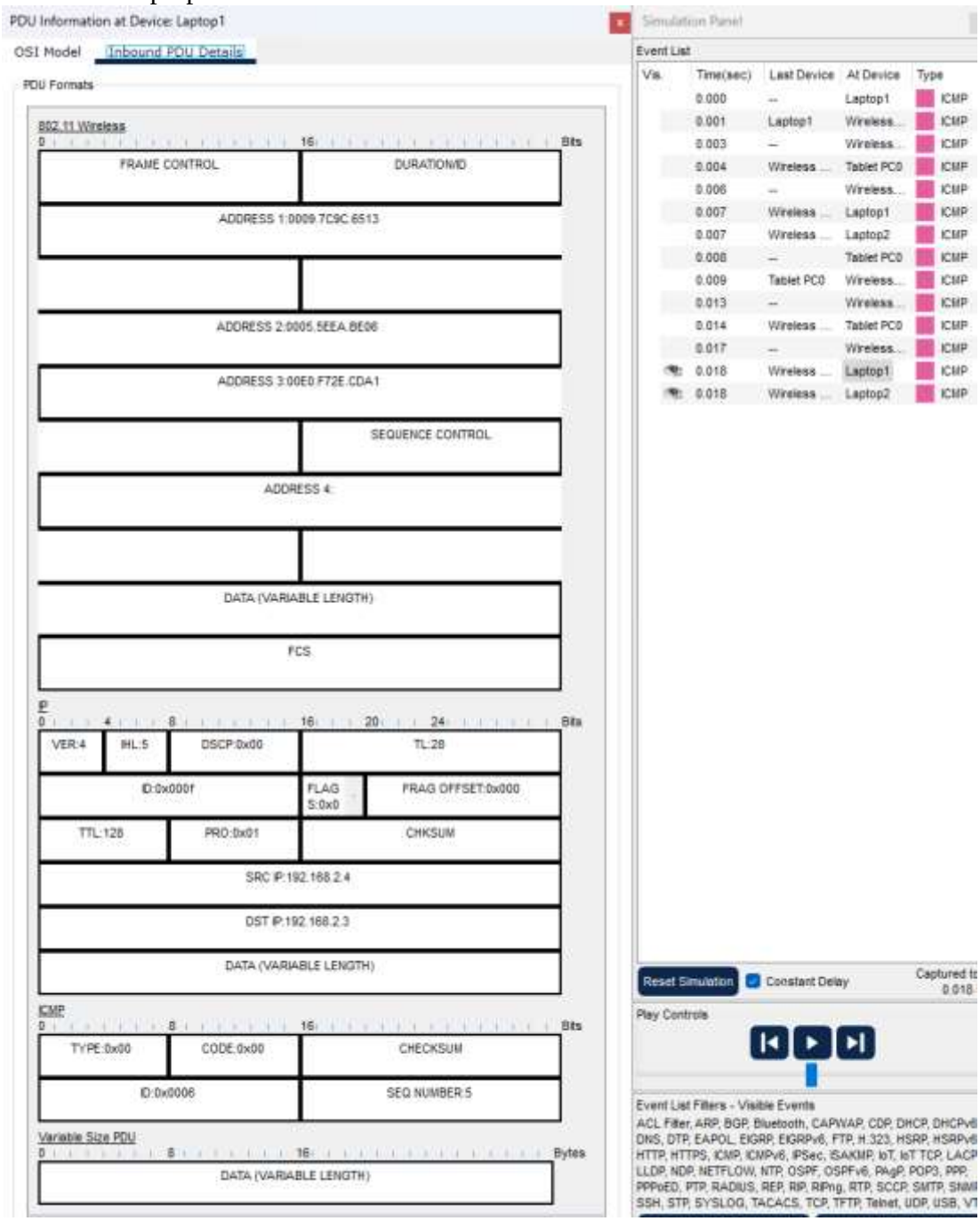


Going to each laptop to connect to the wireless Home Router with the specified SSID

1. Use the simulation mode and send a simple PDU,
    a. From PC1 to PC0



b. From Laptop 1 to Tablet 0



2. Note down the similarities and differences you observed in these to images. You may check the details of PDUs in each device. Explain the reasons behind the differences you observed.

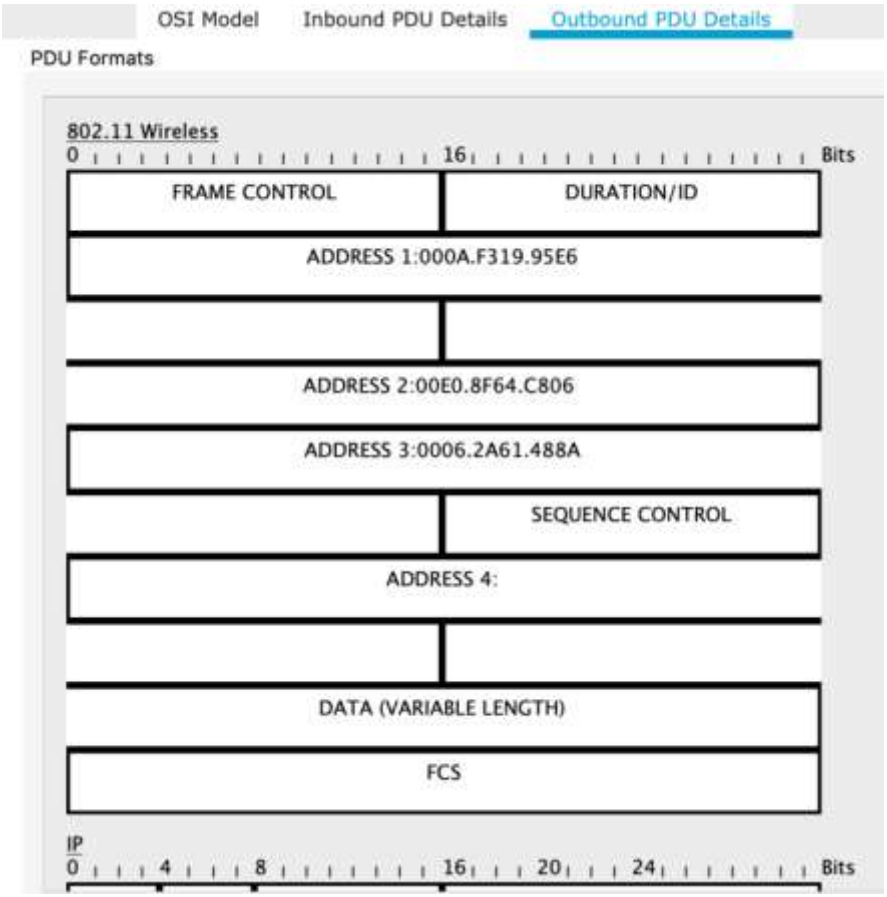
Similarities

- Both of the PDUs use portions of the IP Header by sharing the IPv4 version, header length, and DSCP, TL, and TTL parameters. This is set to 0x01 pointing to the fact that ICMP was used while the ICMP Header uses type 0x00 to show that an ICMP Echo Reply was responded to. In both scenarios there is a checksum present.

Differences

- Since the Data Link layer header of the PDU in Laptop1 has the word "802.11" Frame, which is Wireless Frame and has many address fields meaning there are various layers of addressing in the wireless setup, that means the mode of communication between Laptop1 and Tablet0 is wireless in the Data Link Layer (Layer 2). The communication link between PC1 and PC0 is via wire connection namely Ethernet.

3. Check the details of PDUs in both wired and wireless LANs. The PDUs of wired LAN, you can find source and destination MAC addresses. However, there are three MAC addresses listed in the PDUs of Wireless LAN as shown below.



4. Explain why there are three MAC addresses listed in 802.11 Wireless PDUs?

    - There are three MAC address in in this wireless PDU where;
        - Address 1 is the Destination Address – this is the MAC address of the final destination device.
        - Address 2 is the Source Address – this is the MAC address of the original sender of the packet.
        - The third MAC Address is the Basic Service Set Identifier which is the MAC address of the access point through which the wireless communication is routed.

There is a need to keep abreast three MAC addresses due to the following reasons; While on aspects of wireless network developments, the devices do not actually directly communicate; rather all broadcast communication are relayed through an AP. This leads to a multi hop scenario where a frame need to be routed from a source device to a AP and then to a target device.

Above and Beyond Tasks

There are two types of Ethernet cables used in computer network, i.e., straight through and crossover cables. Explain,

- What is a straight through Ethernet cable? When do we need to use a straight through Ethernet cable?
    - These include the straight through cables which are types of twisted pair cables used in local area network to connect devices like PCs with switches and routers with switches.

- What is a crossover Ethernet cable? When do we need to use a crossover Ethernet cable?
    - PC to PC, switch to switch or even a router to router can be directly connected to each other through a crossover connection.
- How do you identify the type of an Ethernet cable?
    - By the cable color Eth has distinguished between T568A and T568B wiring schedule. A cross over cable is one that implements both wiring patterns while a straight through is one that only implements one.

Task 10.1P: Lesson Review

Module Summary

We worked with a range of fundamental ideas in computer networks and communication during this subject, emphasizing practical application of theoretical knowledge through hands-on exercises. The assignments, which included packet capturing, network configuration, and protocol analysis, gave us a better grasp of how data flows across various network layers. One of the main tasks was to describe the processes a laptop goes through to connect to a website, from turning it on to downloading a page. Important protocols including QUIC for quicker and more secure data transport, ARP for address resolution, DNS for domain name resolution, and DHCP for IP address assignment were found and examined. Together, these protocols—which are all a part of the network stack—assure smooth communication and connectivity.

By utilising Wireshark for packet capture and analysis, we were able to gain even more insight. This made it possible for us to watch real-time protocol exchanges, like how TLS encrypts data to guarantee security, how QUIC enhances traditional connections with reduced latency, and how TCP creates dependable connections through a three-way handshake. The necessity of the transport and security layers in preserving safe, dependable communication between clients and servers was highlighted by the analysis of these collected packets, which offered important insights into the roles that each protocol performs at various OSI model tiers. We also looked at how IP configurations are assigned to devices in a network using the Dynamic Host Configuration Protocol (DHCP) and the Address Resolution Protocol (ARP), which maps IP addresses to MAC addresses.

Additionally, the lesson addressed network implementation using Cisco Packet Tracer, in which we configured a network architecture consisting of wired and wireless local area networks (LANs). We were able to set up wireless access points, setup static IP addresses, and learn about the distinctions between wired and wireless communication thanks to this activity. We learnt more about how wired LANs rely on direct MAC address communication while wireless LANs use more intricate routing through access points and multiple MAC addresses for communication by examining the Protocol Data Units (PDUs) in both types of connections.

Reflecting on the content

The module was very helpful in emphasizing both academic and practical learning. The packet capture analysis using Wireshark, which offered a practical way to comprehend how data moves across a network and how various protocols assure stability and security, was, in my opinion, the most important component. Seeing these protocols in action, such as the three-way handshake of TCP, the quicker data transmission of QUIC, and the encryption of TLS, opened my eyes and helped me see the connection between theoretical ideas and practical uses. My knowledge of networking has increased as a result of this experience, especially with regard to packet analysis, network security, and protocol behavior.

My understanding of networking was more fundamental before this semester, centered on ideas like IP addresses and straightforward network configurations. Nonetheless, this curriculum broadened my knowledge of network architecture and security by introducing me to more complex subjects like TLS and QUIC, along with packet-level analysis. It also made it easier for me to comprehend how crucial it is to use encryption protocols to secure communication and how newer protocols, like QUIC, outperform more established ones.

It's evident that the course team's goal in instructing us on this material is to give us the analytical and practical skills required to debug, examine, and comprehend intricate networking circumstances. A thorough learning experience was guaranteed by the integration of theoretical content with practical assignments. In addition to enhancing my technical abilities, this curriculum has given me insightful knowledge that will be useful in a professional networking setting, particularly in positions involving network management, security, or troubleshooting.

Task 4.1C: Above and Beyond Pass

Application layer

Discussing the concept of TLS and its handshake process based on the Wireshark screen grabs.

- At first, I opened my internet browser and cleared the cache. This helped me to avoid any previous data to mingle with the newly captured packet.

- I launched Wireshark and chose my network interface (such as Wi-Fi) to begin a fresh packet capture.

- I then entered an HTTPS URL, such as https://www.discoverourtown.com, into my browser.

- I ended the Wireshark packet capture when the webpage had fully loaded.

- Evaluate the handshake in TLS.

- I limited my Wireshark view to TLS packets by using the filter tls. I searched for the initial packets used for handshakes, which typically contain:

- The handshake begins with a ClientHello message sent by the client, which is my browser. TLS versions and cipher suites that are supported are examples of this.

- In response, the server selects the TLS version and cipher suite from the client's choices and sends back a ServerHello message.

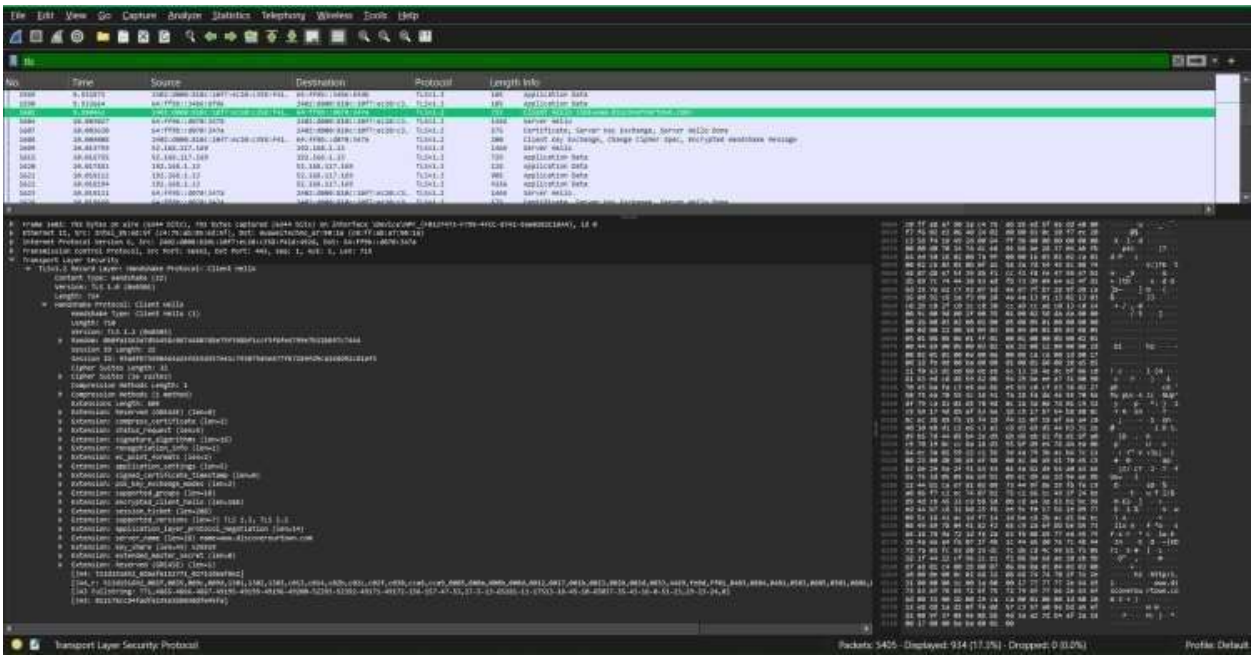
- Certificate: To verify its identity, the client receives a digital certificate from the server.

- The server signals that its portion of the handshake is complete with the message ServerHelloDone.

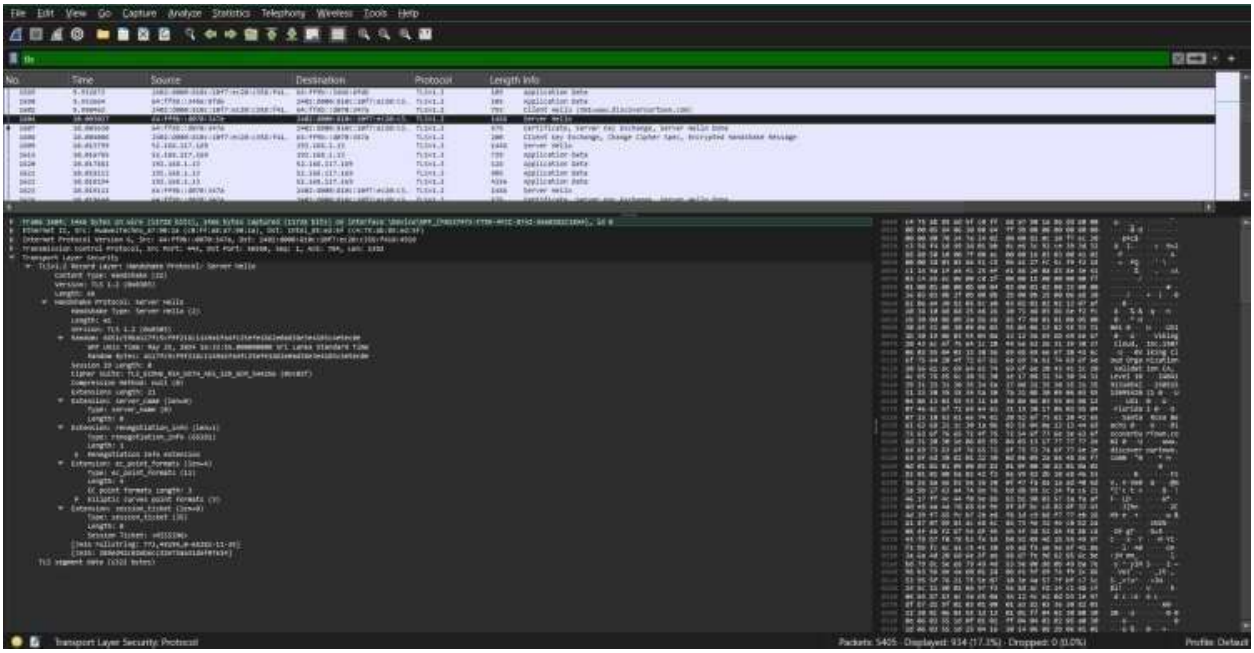
- ClientKeyExchange: Using the public key of the server, the client transmits a pre-master secret that has been encrypted.

- This message is sent by the client and server to transition to encrypted communication (ChangeCipherSpec).

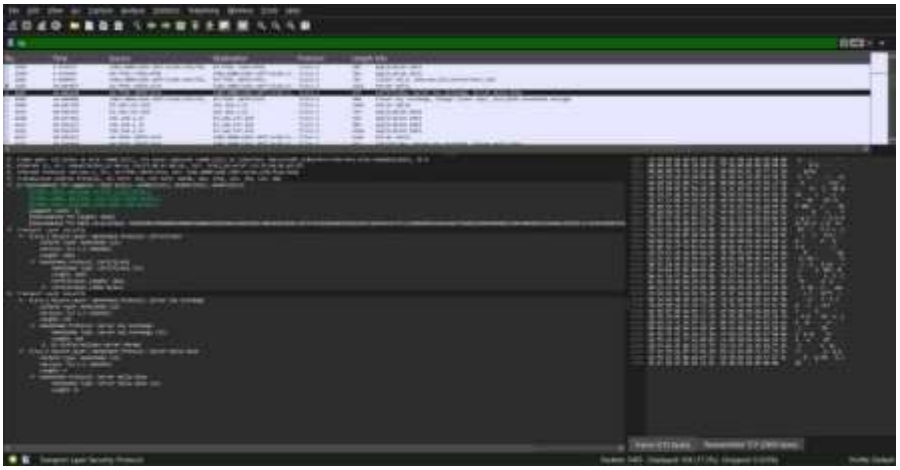
- Completed: To indicate that the handshake is finished, both sides send a completed message.



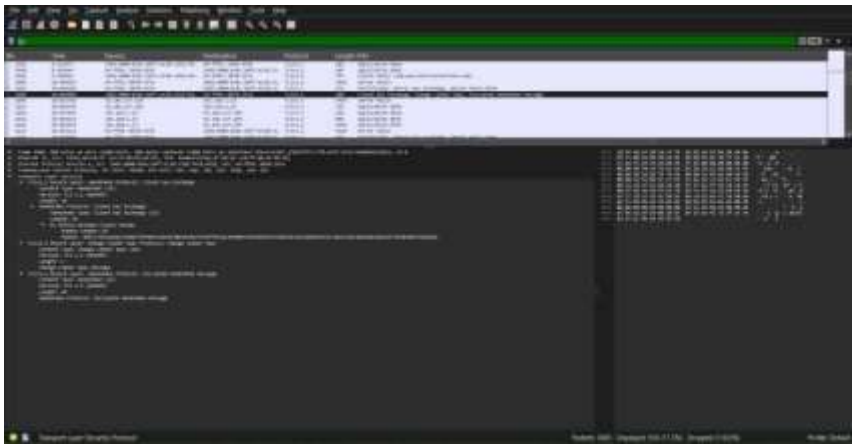
Hello, Client
I located the message ClientHello. It contained information on cipher suites and available TLS versions (such as TLS 1.2 and 1.3).



Hi, Server
The ServerHello message was recognized by me. The server selected the cipher suite and TLS version in addition to sending a random value.



Accreditation
I found the Certificate message indicating that the client had received the server's public key certificate. This was employed to confirm the identification of the server.



Exchange of Client Keys
The ClientKeyExchange message was located. A pre-master secret was encrypted by the client using the server's public key before being transferred to the server.

1. Can you use Wireshark to analyze HTTPS? Give an explanation for your response. If so, give examples of how we can accomplish it. If not, is there another way we might analyze HTTPS without sacrificing security?

- Because HTTPS traffic is encrypted, it is difficult to analyze directly. However, the TLS handshake and the first few unencrypted packets can be recorded by Wireshark.
  As proof, I displayed the TLS handshake packets that I had recorded using Wireshark.

Active Class 3: It's always DNS, No It's 192.168.1.2 (Module 2)

Describe email

- ✓ Email communication mostly uses the Simple Mail Transfer Protocol (SMTP) application layer protocol.
- ✓ For dependable data transfer, SMTP depends on TCP (Transmission Control Protocol), the underlying transport layer protocol.

- The fundamental actions required to send an email from user A to user B are
  - ✓ A Mail User Agent (MUA) such as Outlook, Thunderbird, or a web-based email client is used by User A to write emails.
  - ✓ Using the SMTP protocol, the MUA sends an email by connecting to the SMTP server, which is usually offered by the email service provider.
  - ✓ Using a number of Mail Transfer Agents (MTAs), the SMTP server receives the email and forwards it to the recipient's SMTP server (if different).
  - ✓ The email is received by the recipient's SMTP server, which then saves it in the recipient's mailbox.
  - ✓ With the use of a MUA and protocols like POP3 (Post Office Protocol 3) and IMAP (Internet Message Access Protocol), User B retrieves the email from their mailbox.

- Some of the instructions for sending an email that is defined by the SMTP protocol includes HELO, defining the connection, MAIL FROM defining the sender, RCPT TO, defining the receiver of the message and DATA for sending the body of the message. The two main protocols that are used in getting the emails from the mail server are the POP3, and IMAP. POP3 downloads mails to the local device while IMAP allows for using the mailbox from different devices at a go. Some other parameter of the email are TLS/SSL for security while transmitting the email and DNS (Domain Name System) for identifying the email server address.

Summary for Module 2 active classes 2 and 3

- First, I examined the details of the TLS, a cryptographic system that need to be used to ensure that communications over the internet is secure Secondly, I outlined the TLS handshake, which is a process of creating a secure communication channel between a client and server by negotiating. When deciphering the status of messages that were sent, I employed Wireshark and analyzed packet captures to understand a series of messages that occured during the TLS handshake. That, in turn, helped me gain additional knowledge of the subsequent negotiation of encryption keys and the onset of data encryption as a result. The protocol gave people a good insight into how data transmitted over any network is protected, who authorize it and how that data stays secret.

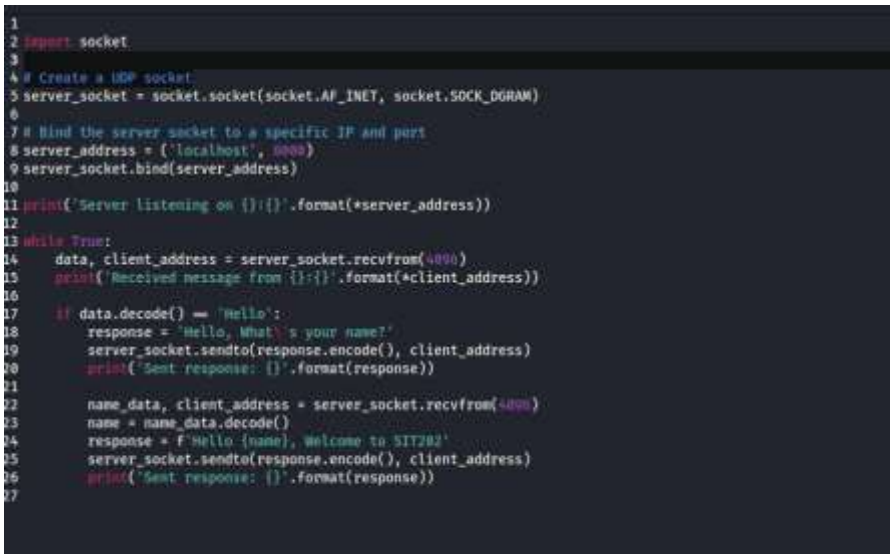
- Besides TLS, I researched how email works since it is one of the most internet essential tools. The basic protocol associated with the sending of emails was the next discussed one which was known as the Simple Mail Transfer Protocol or SMTP for short. We also looked at the design of the Internet and the Transport Control Protocol (TCP) that is used at the transport layer together with the SMTP. It was easy to describe the client-server notion along with the SMTP servers when giving the barest of the steps required to transfer an e-mail message from one user to the other.

Reflection of Module 2 active classes 2 and 3

- It was rather possible to understand the basics of computer communications and Internet security studying the example of email and TLS. Studying TLS handshakes gives an understanding of the machinery behind security connections and shows the importance of the encryption to protect information. From this practical lesson, I enhanced my understanding of the concept of cybersecurity and clearly saw how it is important to use secure encryption measures to prevent cyber-attacks.
- In addition, the email explanation helped untangle the complex process of sending messages through the internet. By being able to consider the process at length I was able to gain insight into how intricate the fundamental infrastructure behind email services are. Thus, this investigation revealed the value of stable protocols such as SMTP and TCP for maintaining user to user free communion regardless of the number of emails services involved.
- Therefore, the analysis of email correspondence and TLS promoted the development of knowledge about protocols at the Internet level and their roles, ensuring the security and productivity of the data exchange process. Through nice of these core concepts, I now have better understanding on how the modern digital communication networks work, allowing me and us, to be more informed as we try to shape the Internet technology realm as it continues to advance.

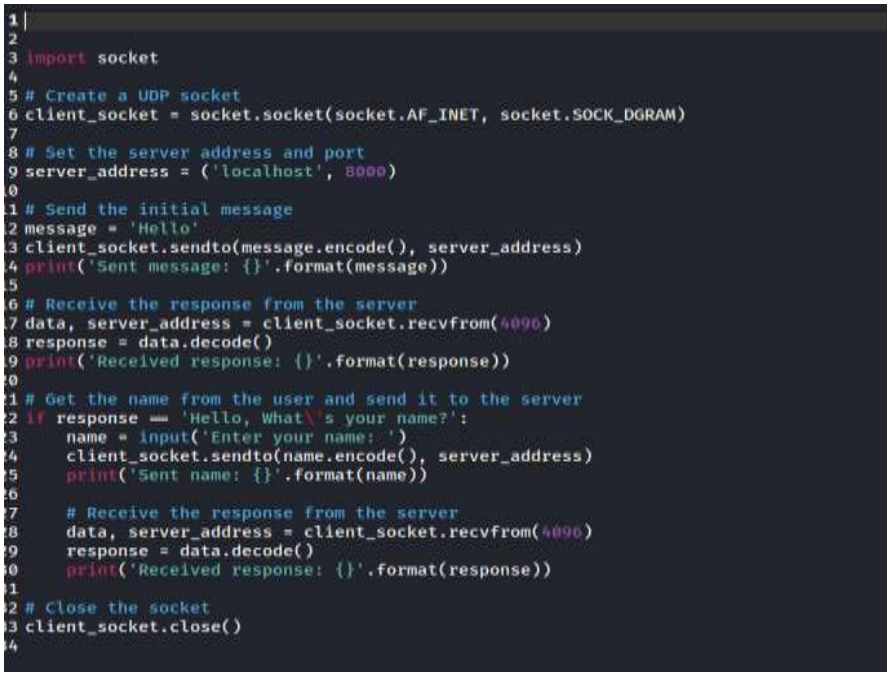
Active class 4: UDP - Unreliable Data Protocol? (Module 3)

Server.py



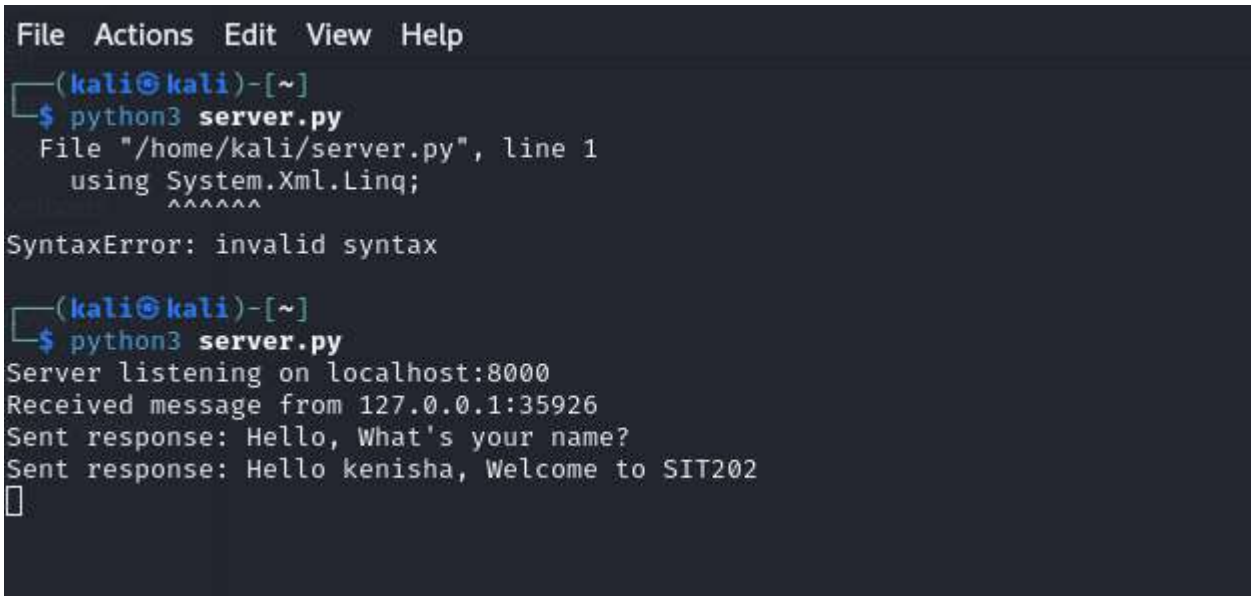
```
import socket

# Create a UDP socket
server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

# Bind the server socket to a specific IP and port
server_address = ('localhost', 8000)
server_socket.bind(server_address)

print('Server listening on {}:{}'.format(*server_address))

while True:
    data, client_address = server_socket.recvfrom(4096)
    print('Received message from {}:{}'.format(*client_address))

    if data.decode() == 'Hello':
        response = 'Hello, What\'s your name?'
        server_socket.sendto(response.encode(), client_address)
        print('Sent response: {}'.format(response))

        name_data, client_address = server_socket.recvfrom(4096)
        name = name_data.decode()
        response = f'Hello {name}, Welcome to SIT202'
        server_socket.sendto(response.encode(), client_address)
        print('Sent response: {}'.format(response))
```

Client.py



```
import socket

# Create a UDP socket
client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

# Set the server address and port
server_address = ('localhost', 8000)

# Send the initial message
message = 'Hello'
client_socket.sendto(message.encode(), server_address)
print('Sent message: {}'.format(message))

# Receive the response from the server
data, server_address = client_socket.recvfrom(4096)
response = data.decode()
print('Received response: {}'.format(response))

# Get the name from the user and send it to the server
if response == 'Hello, What\'s your name?':
    name = input('Enter your name: ')
    client_socket.sendto(name.encode(), server_address)
    print('Sent name: {}'.format(name))

    # Receive the response from the server
    data, server_address = client_socket.recvfrom(4096)
    response = data.decode()
    print('Received response: {}'.format(response))

# Close the socket
client_socket.close()
```

Server Side Terminal



Client side Terminal

Active class 5: How can I transport my application data reliably? (Module 3)

1. **Stop-and-Wait**
   - ✓ In stop and wait, the sender transmits a single packet, and waits before transmitting the next packet till the receiver sends an acknowledgement or ACK. In case the acknowledgment for the packet is not received in a given duration, the packet is resent. This method tends to produce high latency because the sender must wait for an acknowledgment after each packet therefore it may not be efficient.
   - ✓ Look at this as sending one package at a time and not sending the next one until you receive response (like a signed confirmation). In terms of speed, it is rather slow, however the messages' meaning is quite comprehensible. I'm sure folks can well imagine having a delivery person stand by for each and every box! This is good for simple, low-bandwidth situations where slowness isn't an issue it is very important to note that unlike some protocols or systems, the use of this is good for simple, low-bandwidth situations where slowness isn't an issue.

2. **Go-Back-N**
   - ✓ Go-Back-N restricts the maximum number of packets sent over a connection to the window size which allows the sender to transmit a number of packets before receipt of an acknowledgement. The error forcing the sender to retransmit the lost packet and any number of subsequent packets after an error occurrence is a factor. This technique is more efficient than Stop and-Wait but its disadvantage is that if failures occur frequently then several packets will be transmitted unnecessarily.
   - ✓ It means that the delivery person can pick up as many items as the person wishes in this case without a possibility of the decision being denied. The person has to come back and bring all subsequent packages in the unlikely event that the first package was misplaced. While this is a lot faster than stop-and-wait, a high number of packets having been dropped wastes bandwidth. This is a good compromise between simplicity and optimal results for average data transmission.

3. **Selective Repeat**
   - ✓ As with Go-Back-N, Selective Repeat enables a large number of packets to be transmitted before an acknowledgment to them is expected. If any error is detected, only the individual packet that was lost or was in error is resent, not all subsequent packets. Selective Repeat is better than Go-Back-N because this protocol minimizes unnecessary retransmission, which makes it efficient once errors are present. It is somewhat like arranging packages by an intelligent mailbox.
   - ✓ Full advantage can be taken of a postman who brings packages as the mailbox will categorize them. This is the most efficient way because the packets which were not received at the first go will be resent. But the implementation of is the most challenging. This is best in environments where and when there is need to have fast and reliable data transfers and where efficiency is of essence.

Summary for Module 3 active classes 4 and 5

- The change to the Python software was to add onto a prior client-server UDP communication setting I had established. The older client.py and server.py scripts had to be updated in order to allow for a more Speaking as well as Listening in a real time conversation. The words “Hello” and” Hello, what is your name? Were included into the protocol. Greeting are used and then the customer provides the server with his or her name and receives a welcome.

- Concerning the protocols, the discussion focused on how the fundamental three; Stop-and-wait, Go-back-N and Selective Repeat were different from each other. All protocols’ specific working performance features, benefits and drawbacks were considered. I have also questioned which of these is the optimal one in terms of usability, speeds at which the protocols operate and the ability to recover from errors.

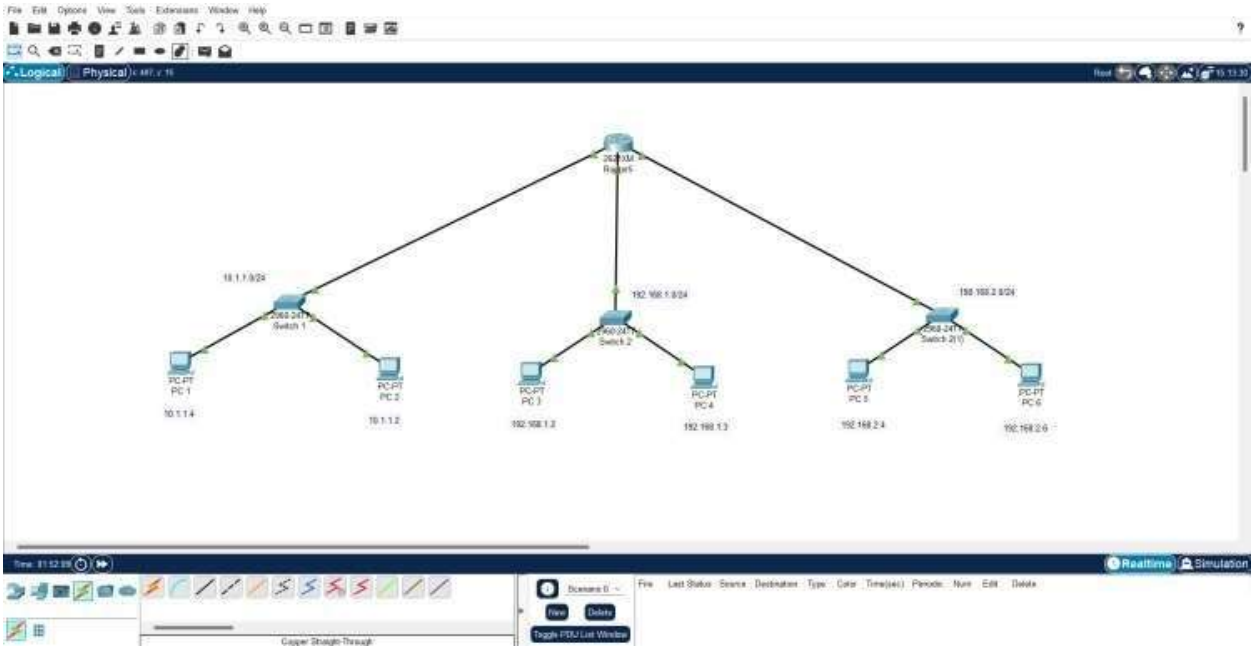
- As will be seen in the subsequent section on TCP congestion control, understanding it was crucial in preserving network stability and reliability. The discussion was centered on role of congestion control technologies in avoiding network congestion and how such technologies are important in issues to do with resource allocation and system failure. While, the congestion control mechanism of TCP includes one of its flexible features since it is capable of changing the transmission rates proportional to the network conditions.

Reflection of Module 3 active classes 4 and 5

- Replacing Python programs offered valuable information of the subject of sockets through stressing the client-server relation and message transmitting techniques. Although this was a shallow practice exercise, comprehension was increased by constructing a basic network communication system through the implementation of UDP sockets. Therefore, by analyzing these protocols for their concrete parameters and uses in practice, the authors’ comprehension of them was improved as well. This way, understanding the strengths and weaknesses of the protocol, and comparing the two and contrasting, this study was able to provide a better appreciation of its effectiveness and inefficiencies primarily on how important it is to choose the right protocol for a particular form of communication needs.

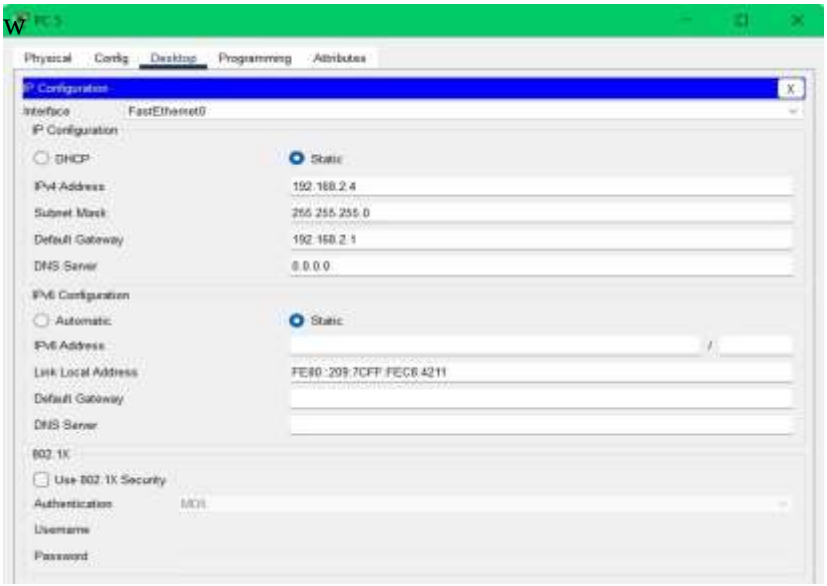
- Apart from filling the knowledge gaps about TCP congestion control, it outlined the difficulties inherent to resource management in distributed settings. Enhanced comprehension of the principles of network communication result from analyzing the congestion control algorithms to explain the role of TCP in managing reliable and highly efficient data flow throughout networks.

Active class 6: How do I get from My home to Your home - Journey of an IP packet (Module 4)

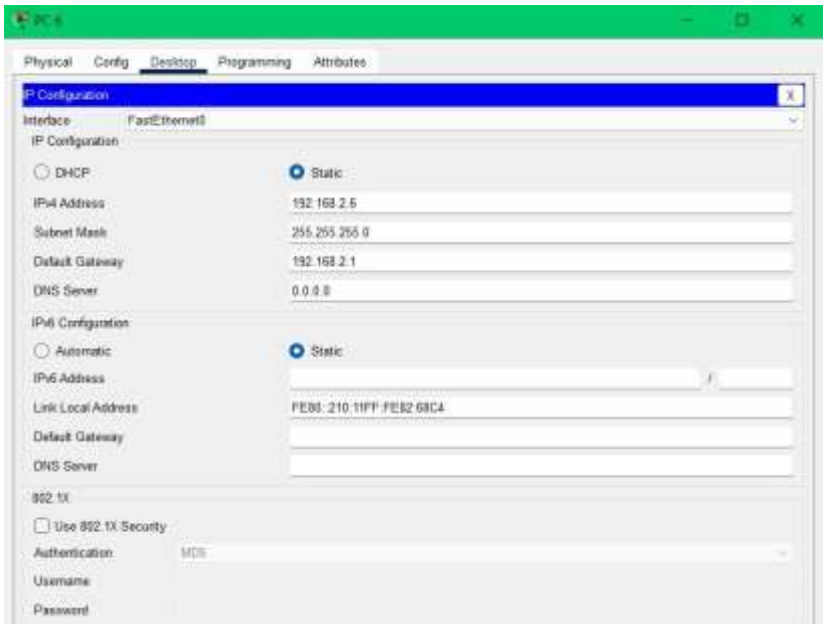


- PC5 and PC6 were connected to the newly formed LAN by me. Assigned to it is the subnet mask 192.168.2.0/24. The following is the LAN3 configurations that we have: I configured the new LAN with the IP addresses as a static one. According to the assignment, the IP address 192.168.2.4 was assigned to the PC5.

PC5

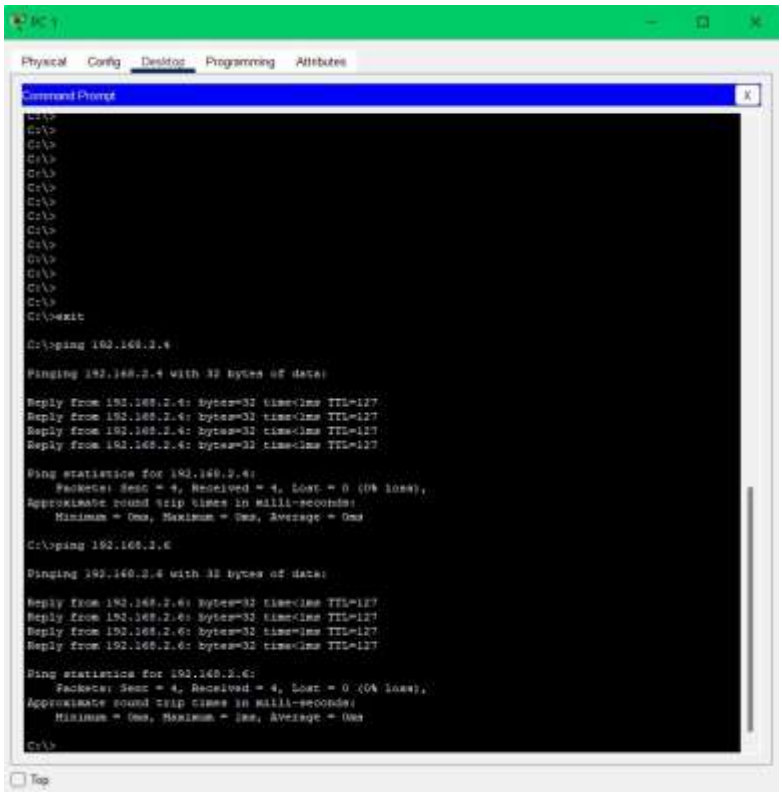


PC6

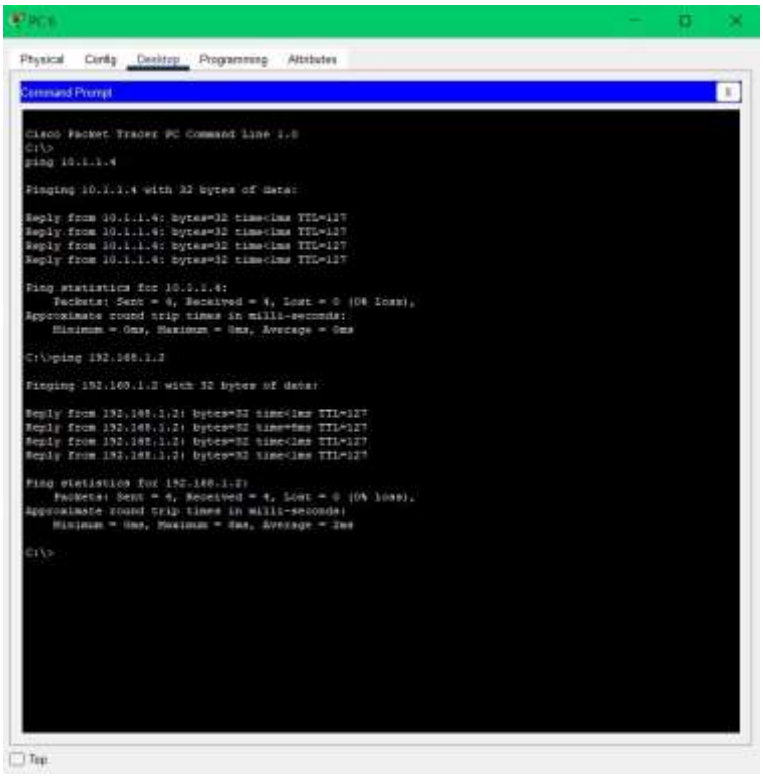


- ✓ To ascertain a connectivity to this LAN3, I had to add more switch and change a router from 1941 to 2621XM. I had to add the NM-1E after switching the router because it has one 10BaseT Ethernet port that can be used to connect a LAN backbone for supporting 24 synchronous/asynchronous ports or six PRI connecting to ISDN lines. Furthermore, I also plugged in the PC6 into the LAN 3 network.

Here's ping from PC1 to PC5 and then from PC1 to PC6



To ensure connectivity from LAN3 to other LANS, I pinged from PC6(LAN3) to PC1(LAN1) and PC3(LAN2)



Summary for Module 4 active class 6

- In order to perform the “Above and Beyond” activity for Active Class 6, I added a new PC into the network created during Activity 3 connecting PC5 with the IP address 198.168.2.4. To achieve this, the current network connectivity has to be reflected on the IP address of PC5. Moreover, I employ the use of the “ping” tool to check whether the PC1 and PC5 were well connected. But for this experiment I was able to have live practical experience in extending the network capabilities As a result of this venture I was able to understand IP addressing and network topology at a deeper level.

Reflection of Module 4 active class 6

- This assignment also offered participants a chance to apply and build on the networking concepts identified in the Active Class 6 activities. Performing an addition of devices and checking the network connectivity also helped to enhance my knowledge on the issue of IP addressing, subnetting as well as the communication protocols in the network. This task required me to put my critical and analytical skills into operation while I systematically tried to understand the structure of network architecture as well as its configuration. Taking everything into consideration, the above and Beyond challenge was a good practice of the actual operation and made me better understand the network layer.

Summary and reflection of Module 2

Summary

- In Module 2, I learned different network application architectures include the Peer-to-Peer (P2P) and Client-Server models. I learned their issues, channels of passing information and information structures. Web services and databases have enormous advantages in Client-Server architecture focusing the centralized control for clients to request services. P2P networks, on the other hand, distribute work among the peers, thus offering scalability at the same time that they pose administrative problems.

- The lesson dedicated significant time to talking about sockets and interprocess communication (IPC), or the way by which processes in a network can convey with one another. Another aspect was the HTTP protocol, which rules client-server communication in the internet space. The important ideas which were highlighted were the effects they have on the performance of connection over the internet, HTTP on the other hand is stated less as being stateless, the differences between persistent and non-permanent connections.

- It also included the Domain Name System (DNS) and its structure, and basic concept of Recursive and iterative approach in DNS. Network safety was also discussed in detail including aspect of encryption and protection against threats including DNS spoofing and man-in-the-middle attacks.

Reflection

- By the time I was editing the notebook, I understood that the lessons on HTTP protocol, Client-Server architectures, and P2P were rather helpful. I benefited from the knowledge I acquired on host communication and data transmission in the module which I have background information from. Enhanced my technical competency of the job connected the gap of theory and practical by doing works of live projects like socket programming and DNS setting. In consideration of all the benefits and drawbacks, the module offered a good foundation for subsequent research in networking, security and application.

Module 3 Summary and Reflection

Summary

- I gained a clear understanding of the TCP/IP idea in module 3, with reference to the Transport Layer role of delivering reliable data delivery between the networks. After comparing between UDP and TCP, I found that TCP guarantees reliable transmission through connection-oriented communication it employs flow management and congestion control and checking for transmission errors. However, UDP protocol offers connectionless services at higher speeds, making it ideal for use in real time clients that do not necessitate regular connection reliability such as on streaming services.

- In as much as the two elements were learned in the module, the distinctions between the Transport and Network Layers were also stressed. Process-to-process communication is designed by the Transport Layer and the delivery of the packets for the hosts is done through the Network Layer. I then considered connections in TCP and more explicitly the three ways handshake to establish a connection and the four ways handshake to close it. Also, I learned features of multiplexing and demultiplexing, which help in managing the communication flow between diverse applications.

- I realized from the practical demonstrations that while applications that require data integrity such as web browsing use TCP, applications that require speed such as online gaming or video streaming use UDP. Another lesson learned was getting to understand how TCP on manages retransmissions and how it comes up with the round-trip time (RTT) so as to be in a position to determine how networks manage to handle packet and delays.

Reflection

- Reflecting on the practice from Module 3, the most important thing to bear in mind was to grasp multiple roles that TCP and UDP have in data transfer thoroughly. I now have a better appreciation of how power flow control, congestion control, and retransmission of TCP is to maintaining data integrity used in programmes like file transfer programs as well as browsing. This is really significant when each packet should reach the target destination on time and in sequence. However, because the UDP puts more importance on conceiving speed with efficacy than on the costs of establishing the connection, it is most valuable in the real-time application such as gaming and streaming whereby a few talk loss is tolerable.

- As for my previous knowledge on the OSI model and this module's theme, which concerned the function of the transport layer in end-to-end communication, those concepts were overlapping significantly. In particular, it facilitated my level of grasping how discovered theoretical concepts as far as network communication is concerned, such as connection-oriented and connectionless, manifest in practical applications. Real life scenarios which were provided here like three way handshake of TCP or using UDP for video streaming gave me a clear understanding of these protocols.

- In my opinion the course team concentrated on this material for the reason that anybody who is attending network engineering or similar courses would require knowledge of, has to have some understanding of the transport layer. Writing networked applications always meant understanding how to receive and send trustworthy data, as well as how to work around performance and network problems. Also, such elements as round-trip time computations and demultiplexing along with multiplexing have enriched the set of the problems I can solve and prepared me for further networking subjects.

- Altogether, this module enabled me to fill the gap of the knowledge difference between book learning and the practice by providing me with the necessary background information and practical skills required to work with the network protocols in real life situations. I feel much more confident now, knowing how to approach the problems of the network and learning how to optimize the communication and ensure that the delivered services are both reliable and efficient. With such an understanding of networking and application development continuing to improve this is the first step in their better comprehension to me.

Summary and Reflection of Module 4

Summary

- Every Internet-connected device contains a network layer used to send and receive segments from a broadcasting host to a receiving host. In this Unit, I examined specific processes which are inherent in these steps in the Module 4. The two main network operations which were discussed in this module were routing and forwarding. Routing involves employing route finding techniques such as BGP and OSPF to determine the best path by which packets should take in order to get from source to the destination. Forwarding is the technique of forwarding the packet from input of the router to the output connection within a locale.

- Two other areas I explored were the data plane and the control plane. On one hand, the control plane is solely responsible for decision and network routing logic whereas on the other hand, it concerns with the actual forwarding of the incoming packets as well as the flow of traffic. It was useful for the module to understand a couple of network service models like Best-Effort Service whose delivery time, delivery order, or bandwidth is not guaranteed.

- Additionally, design of routers was discussed whereby internal components such as input ports and switches, switching fabrics, and the output ports were described in detail regarding how routers use them to manage receipt of packets, switches and transmission of packets. Indeed, the role of subnetting and IP addressing— activities that involves breaking the large network into small sub-networks as noted in the module by assigning and managing the IP addresses with the help of the subnet masks and CIDR notation— was stressed.

- I also learned how NAT works, through which a large group of inside devices can use one public IP address, and DHCP which is used to assign IP addresses dynamically. Also I researched on IPv6 as a possible replacement to the IPv4 address noting a use of 128 bit addressing, and increases in network management and routing.

- Finally, the module explained how this great advance in the entire IP address management system, better problem solving techniques, and efficient transference of the data. Among the topics presented as important for network architecture optimization, connectivity maintenance, and networking future preparedness were subnetting, DHCP, NAT and IPv6.

Reflection

- In this module, the most important thing I learnt was how the data plane functions within the network layer; how routers work, particularly in terms of packet forwarding. Now I am aware of the essential difference in speed between the control plane, which runs in Milliseconds and the data plane which runs in Nano seconds. I was able to enhance the understanding of how I can confidently explain and manage efficient network performance and data transmission.

- This subject added to my previous knowledge of computer networks by introducing me to CIDR and subnetting, which are vital in the tackling of the IP address and mapping of the network. In some ways I feel that I now fully understand how to apply the feature of DHCP for dynamic IP allocation than before and also understand how to implement subnetting in the right manner.

- The course team may have paid particular attention on this content because it provides good background information with regards to networks, which is central to debugging and optimizing their performance. This means that it is important to have adequate knowledge of these components such as DHCP, NAT and IPv6 in order to create safe and efficient networks and especially for home and business use.

- Thus, regarding myself, I found the practical sessions containing the acting out scenes rather useful. While performing the laboratory assignments, I such things as routing tables, analyzing the packet flows and configuring the subnet masks, to help augment the theory learnt. Another problem I faced was with subnetting at the initial stages but I managed to overcome this with some practice. These encounters emphasized an importance of having detailed focus on the situation, given that small changes in configuration of the network can affect its function significantly.

- Thus, this session really has equipped me with the basic technical knowhow, required to overcome any networking head ache in future. The knowledge of IP address management, routers and other aspects within the network layer shall be quite helpful when it entails attaining and implementing network solutions for security and high performance.

Task 5.2C My DNS Server Sketch

Task

1. Server Initialization: Outline the steps for starting a DNS server, including initialization of necessary components.

    - **Server initialization**
      In this stage, the required parts are also assembled, and the DNS server is set up. DNS typically operates using UDP for the fast handling of its queries; therefore, the server uses the User Datagram Protocol (UDP). Here the server socket is bound to the normal DNS port which is 53.

```
# Function to initialize the DNS server
def initialize_server():
    # Create a UDP socket
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    # Bind to the local address (IP) and port
    try:
        server_socket.bind(("0.0.0.0", 53))  # Bind to port 53 (DNS)
        print("DNS server started and listening on port 53")
    except:
        print("Failed to bind to port 53")
        sys.exit(1)
    return server_socket
```

    - **Explanation**
      UDP Socket Creation: To create a UDP socket for network connection the server uses socket.socket(socket.AF_INET, socket.SOCK_DGRAM).
      Port Binding: In the same way that server_socket.bind(("0.0.0.0", 53)) makes the server listen for DNS queries on port 53, server = socketserver.TCPServer(("", 53), DNSHandler) does the same.
      Error Handling: If the server does not bind to port 53 then the server quits with an error message.

2. Listening and Processing DNS Queries: Describe how the server listens for and processes incoming DNS queries. Develop logic for parsing queries to identify the hostname and query type (A or CNAME).

- **Listening and Processing DNS Queries**
  Once the server has been started up, it listens for DNS requests that are coming in. The flags used in determining a transaction ID, the hostname to which the response belongs and query type (A or CNAME) can all be gotten from analyzing the DNS message that is received.

```
# Function to listen for incoming DNS queries
def listen_for_queries(server_socket):
    while True:
        # Receive a DNS query (max buffer size of 512 bytes)
        query, client_address = server_socket.recvfrom(512)ww
        print(f"Received query from {client_address}")
        # Parse the DNS query to extract the transaction ID, flags, hostname, and query type
        transaction_id, flags, questions, query_name, query_type = parse_dns_query(query)
        # Handle the query based on the type (A or CNAME)
        if query_type == "A":
            handle_a_record(query_name, client_address, transaction_id, server_socket)
        elif query_type == "CNAME":
            handle_cname_record(query_name, client_address, transaction_id, server_socket)
        else:
            send_error_response(client_address, transaction_id, server_socket)
```

- **Explanation**
  Receiving enquiries: The recvfrom function (512) is used by the server to listening to the enquiry and to collect up to 512 bytes of information from the client.
  Parsing Queries: parse_dns_query(query) returns the hostname and query type (A or CNAME), and many other essential pieces of information.
  Conditional Handling: For CNAME records, the server supports handle cname record whereas for A records, it can handle a record only.

3. Handling A and CNAME Records: Design separates logical flows for dealing with A and CNAME record queries.

- **Handling A and CNAME Records**
  For such reasons, the server uses different routes of logic to address A and CNAME record searches. For CNAME records, it gets the alias while for A records it gets the address.

```
# Function to handle A record queries

def handle_a_record(query_name, client_address, transaction_id, server_socket):

    dns_records = {

        "example.com": {"A": "93.184.216.34"},

        "test.com": {"A": "192.0.2.1"}

    }

    if query_name in dns_records and "A" in dns_records[query_name]:

        ip_address = dns_records[query_name]["A"]
        response = generate_dns_response(transaction_id, query_name, "A", ip_address)
        server_socket.sendto(response, client_address)
        print(f"Sent A record response to {client_address}")

    else:

        send_error_response(client_address, transaction_id, server_socket)

# Function to handle CNAME record queries
def handle_cname_record(query_name, client_address, transaction_id, server_socket):

    dns_records = {

        "alias.com": {"CNAME": "example.com"},
        "anotheralias.com": {"CNAME": "test.com"}

    }

    if query_name in dns_records and "CNAME" in dns_records[query_name]:

        cname_value = dns_records[query_name]["CNAME"]
        response = generate_dns_response(transaction_id, query_name, "CNAME", cname_value)
        server_socket.sendto(response, client_address)
        print(f"Sent CNAME record response to {client_address}")

    else:

        send_error_response(client_address, transaction_id, server_socket)
```

- **Explanation**
  Handling A Records: If the questioned domain contains an A record, the server get the associated IP address and launches the DNS response.

  Managing CNAME Records: When it comes to CNAME queries, the server pull off the alias domain, (CNAME) and generate the proper DNS response.

  Error Handling: When the domain is not found with the record database or is not valid, It results the server to return an error response.

4. Generating DNS Responses: Develop the process for creating and sending appropriate DNS response messages

    - **Generating DNS Responses**
      Last but not least, depending on the type of the query – it could be query type A or CNAME, the server then comes up with the DNS answer. There is a header, the questions' part and the answer part in the reply.

Answer Header

```
# Function to generate a DNS response
def generate_dns_response(transaction_id, query_name, record_type, record_value):
    # Construct the response header
    flags = b'\x81\x80'  # Standard query response, no error
    question_count = b'\x00\x01'  # One question
    answer_count = b'\x00\x01'    # One answer
    ns_count = b'\x00\x00'        # No authority records
    ar_count = b'\x00\x00'        # No additional records

    # Construct the question section (same as the query)
    question_section = generate_query_section(query_name, record_type)

    # Construct the answer section (A or CNAME)
    answer_section = generate_answer_section(query_name, record_type, record_value)

    # Return the full DNS response
    return transaction_id + flags + question_count + answer_count + ns_count + ar_count +
question_section + answer_section
```

Section Answer

```
# Function to generate the answer section for A or CNAME
def generate_answer_section(query_name, record_type, record_value):
    pointer_to_query_name = b'\xC0\x0C'  # Use label compression

    if record_type == "A":
        answer_type = b'\x00\x01'  # A record
        answer_value = socket.inet_aton(record_value)  # Convert IP to binary
    elif record_type == "CNAME":
        answer_type = b'\x00\x05'  # CNAME record
        answer_value = convert_name_to_dns_format(record_value)

    answer_class = b'\x00\x01'  # Class IN (Internet)
    ttl = struct.pack('!I', 300)  # 5 minutes TTL
    data_length = struct.pack('!H', len(answer_value))  # Length of the answer

    return pointer_to_query_name + answer_type + answer_class + ttl + data_length +
answer_value
```

- **Explanation**
  Answer Header: The header (flags = b'\x81\80') has proved one question and one answer of the last successful query.

  Section of Answer: In answer part the server gives the IP address (A record) or an alias (CNAME) if it is an A or CNAME request.

Task 6.2C My DNS Server

Step 1

- ✓ The first step I took was to review well the work criteria to ensure that I had the best understanding of what was required of me. The required steps were to create DNS server and client only in python and without using any pre-made DNS packages. Therefore, the specific requirements are: 1 to support Hostname-to-IP-address translation, 2 host aliasing, 3 either A or CNAME DNS record types, and 4 it has to use UDP.

Step 2

- ✓ I first built the map for I then had to set it aside for the purpose of storing the DNS records. This dictionary contained the following information: the used hostname, the corresponding value and, finally, the type of DNS record – A or CNAME. An alias.com record would be 'alias.com': An MX record would be example.com : {‘type’: ‘MX’, ‘value’: ‘mail.example.com’} while an A record is example.com : {‘type’: ‘A’, ‘value’ :’93.184.216.34’}.

Step 3

When I wrote the DNS server code, I made sure it included the following features:

- ✓ Socket Creation: To achieve this I used second variable called data_socket which is a UDP socket and I used the bind method from socket module of Python to bind the socket to the given port and address. Conflicts are another recurring consideration and to minimize them, I initially choose my DNS port to be 53 and later on switched to 8053.
- ✓ Handling queries: In order to handle petitions I established a handle_query method that from an HList decomposes the query, decoded it in order to obtain the hostname and the type of record requested and proceed to verifying the records of the DNS and configured a way to reply. In case the CNAME was followed by an A record the server replaced the CNAME with the A record IP address.
- ✓ Listening for Inquiries: The waiter eagerly waited for new questions to pop up. It addressed each one using the handle query method and returned the relevant result.

Step 4

I then developed the DNS client code that manipulates the server.

- ✓ Sending Queries: The client then passed the host name and record type entered by the user to the server through a UDP socket after setting the query.
- ✓ Receiving Responses: The query was provided by the server and after it, the client was shown the information by the user.
- ✓ User Interaction: The client continued to ask the user questions until the user gave BlackBox a command to cease its questions.

Step 5

When I ran the server code, I noticed that Port 53 was already in use and that I was getting an OSError: [Errno 98] Address already in use. To correct this:

- ✓ Port Modification: The port number 8053 for the server was modified and the start_dns_server function was also modified.
- ✓ Client Update: The client code was modified to meet the new server port number being 8053.

Step 6

I used the server and client to test the implementation:

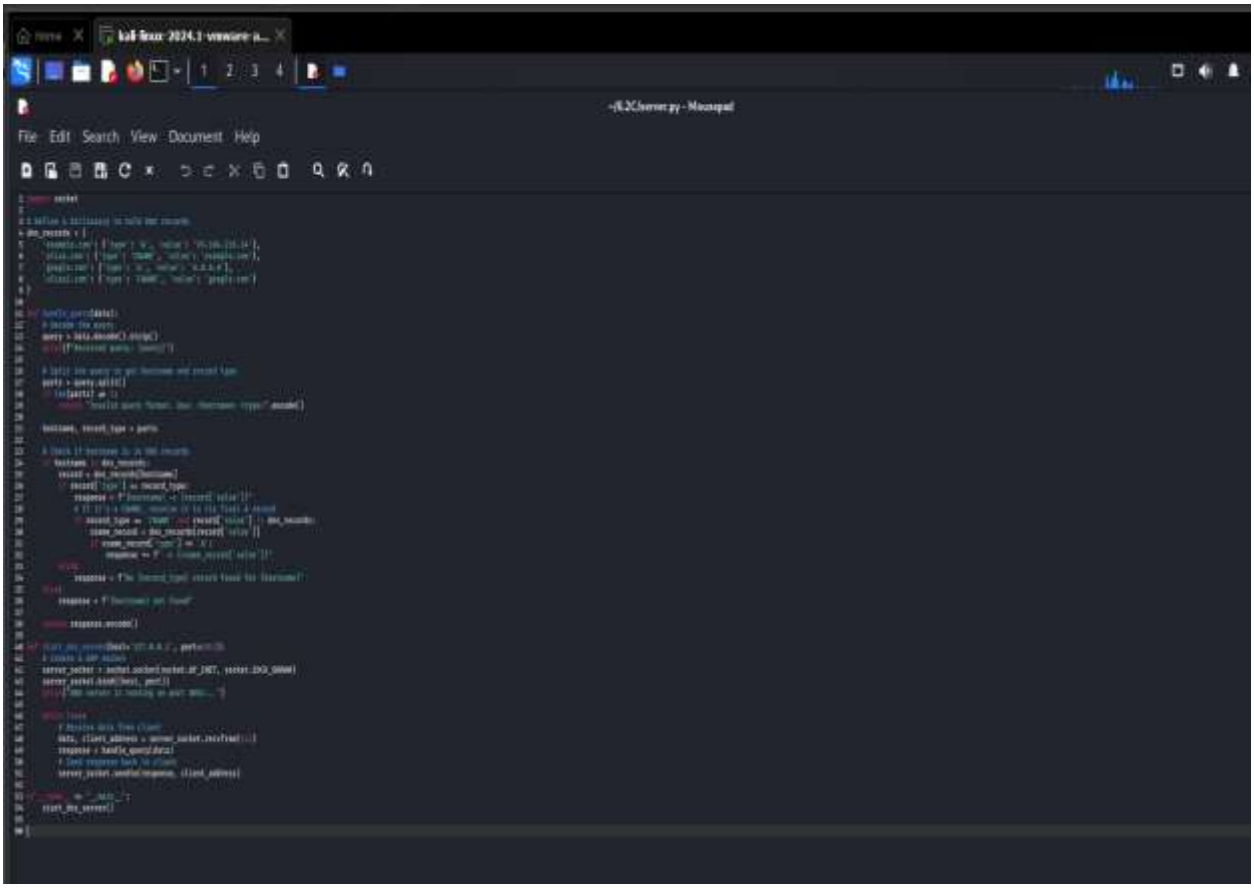
- ✓ Launching the Server: To ensure that the server was running on port 8053 I ran the code dns_server.py.
- ✓ Starting the Client: I ran dns_client.py to the main from shell, entered the hostnames and record kinds and checked if the client is receiving correct responses from the server.

I'll now give you the client and server codes.

Server code

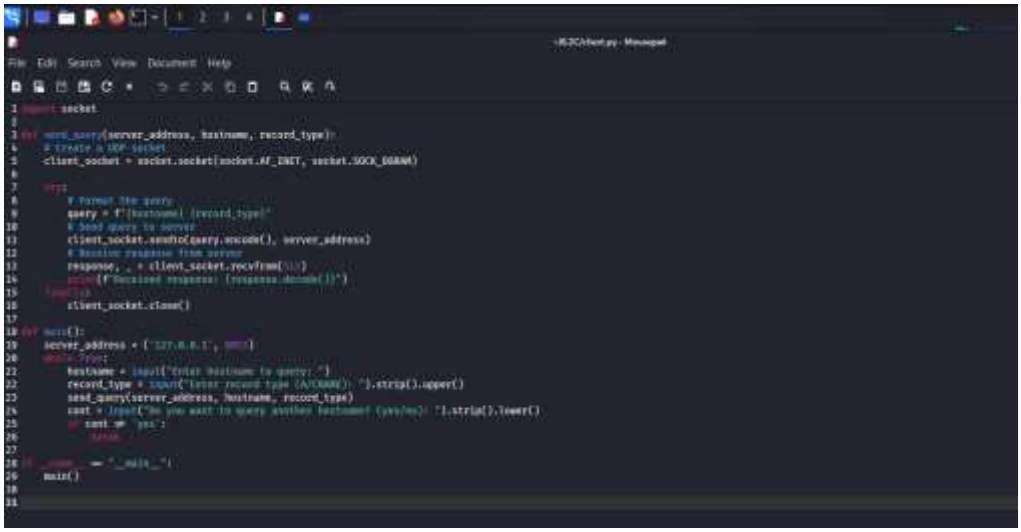
```
import socket

# Define a dictionary to hold DNS records
dns_records = {
    'example.com': {'type': 'A', 'value': '93.184.216.34'},
    'alias.com': {'type': 'CNAME', 'value': 'example.com'},
    'google.com': {'type': 'A', 'value': '8.8.8.8'},
    'alias2.com': {'type': 'CNAME', 'value': 'google.com'}
}

def handle_query(data):
    # Decode the query
    query = data.decode().strip()
    print(f"Received query: {query}")

    # Split the query to get hostname and record type
    parts = query.split()
    if len(parts) != 2:
        return "Invalid query format. Use: <hostname> <type>".encode()

    hostname, record_type = parts

    # Check if hostname is in DNS records
    if hostname in dns_records:
        record = dns_records[hostname]
        if record['type'] == record_type:
            response = f"{hostname} -> {record['value']}"
            # If it's a CNAME, resolve it to its final A record
            if record_type == 'CNAME' and record['value'] in dns_records:
                cname_record = dns_records[record['value']]
                if cname_record['type'] == 'A':
                    response += f" -> {cname_record['value']}"
        else:
            response = f"No {record_type} record found for {hostname}"
    else:
        response = f"{hostname} not found"

    return response.encode()

def start_dns_server(host='127.0.0.1', port=8053):
    # Create a UDP socket
```

```
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    server_socket.bind((host, port))
    print("DNS server is running on port 8053...")

    while True:
        # Receive data from client
        data, client_address = server_socket.recvfrom(512)
        response = handle_query(data)
        # Send response back to client
        server_socket.sendto(response, client_address)

if __name__ == "__main__":
    start_dns_server()
```



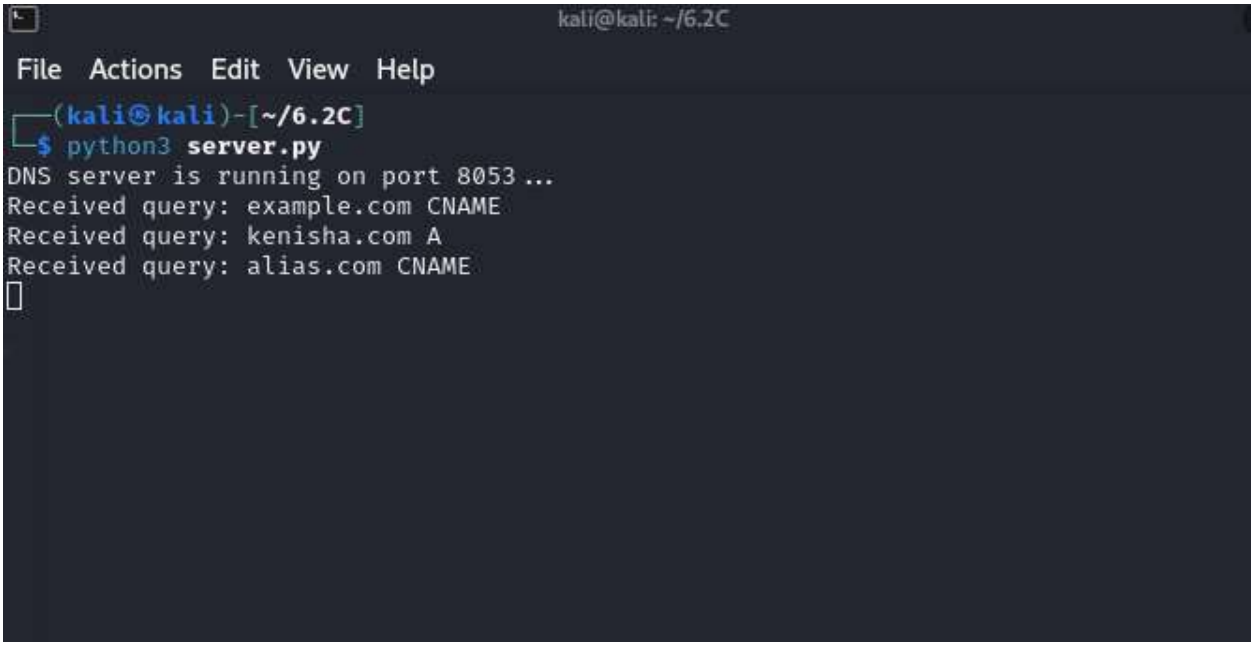
Client Code

```
import socket

def send_query(server_address, hostname, record_type):
    # Create a UDP socket
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    try:
        # Format the query
        query = f"{hostname} {record_type}"
        # Send query to server
        client_socket.sendto(query.encode(), server_address)
        # Receive response from server
        response, _ = client_socket.recvfrom(512)
        print(f"Received response: {response.decode()}")
    finally:
        client_socket.close()

def main():
    server_address = ('127.0.0.1', 8053)
    while True:
        hostname = input("Enter hostname to query: ")
        record_type = input("Enter record type (A/CNAME): ").strip().upper()
        send_query(server_address, hostname, record_type)
        cont = input("Do you want to query another hostname? (yes/no): ").strip().lower()
        if cont != 'yes':
            break

if __name__ == "__main__":
    main()
```

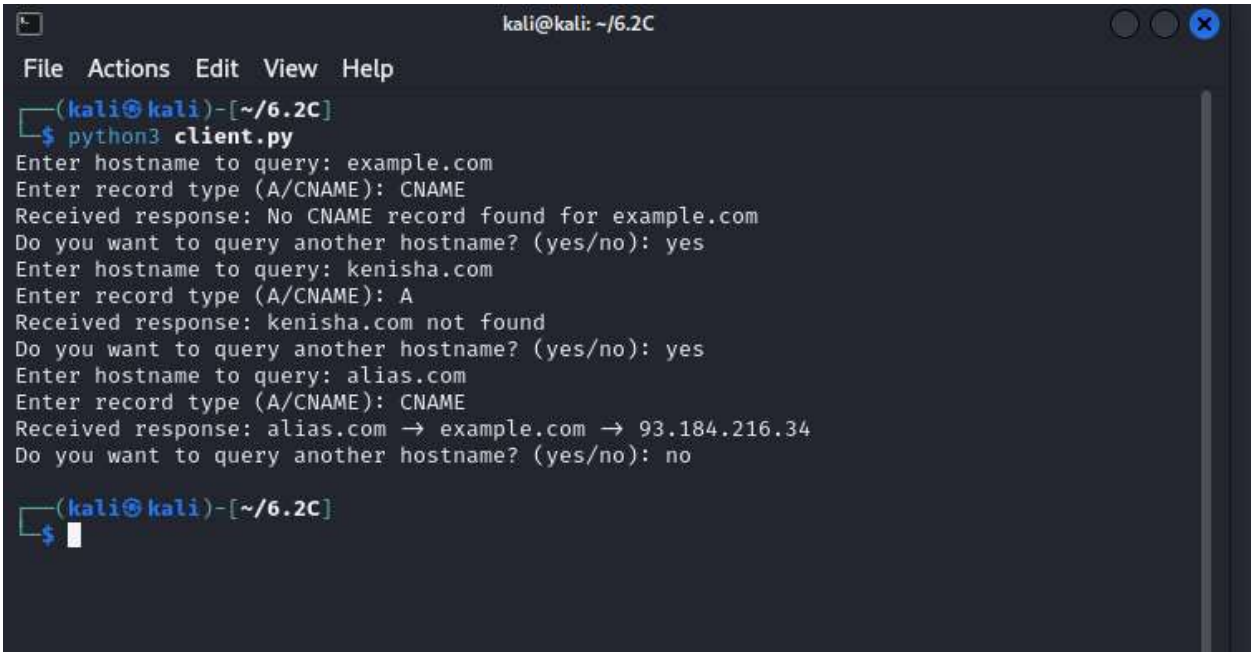


I will now present the results of both codes.

Sever Output



Client Output



Summary and reflection on this Task

Summary

- In this assignment, I developed a DNS server and client program on the Python programming environment. This was the goal of constructing a hypothetical DNS server capable of handling resource records of CNAME (canonical name) and A. The server required to answer queries over UDP, the client had to be able to ask questions to the server and receive answers in return.

- The DNS server doubles for questions coming its way by parsing the hostname and the record type, searching for the records and providing the appropriate response. Instead, the DNS server maintains DNS records in a dictionary. Both the hostname and record type were provided by the user and input into the client program and the request was sent to the server and the result was displayed.

- An error occurred: [Errno 98] When I first began getting problems with the default DNS port which is 53, the message showing was ‘address already in use’. As a solution, I changed the server and client to use port 8053. By so doing, client-server interaction was enhanced and servers run effectively without conflicts.

Reflection

- On balance, it was a pleasant practice in terms of accomplishing educational objectives of the work. First of all I extended my knowledge with DNS protocols especially with the usage of UDP for DNS queries and responses. Since I had to implement the server part and the client part using my own algorithms not relying on any other DNS libraries it was quite difficult for me to get a complete grasp of how DNS operations work in details.

- After the port conflict issue was discovered and resolved, some of the activities among them were trouble shooting and problem solving. It emphasized the fact that more can be learned about the network settings and some leeway should always be provided since problems may occur.

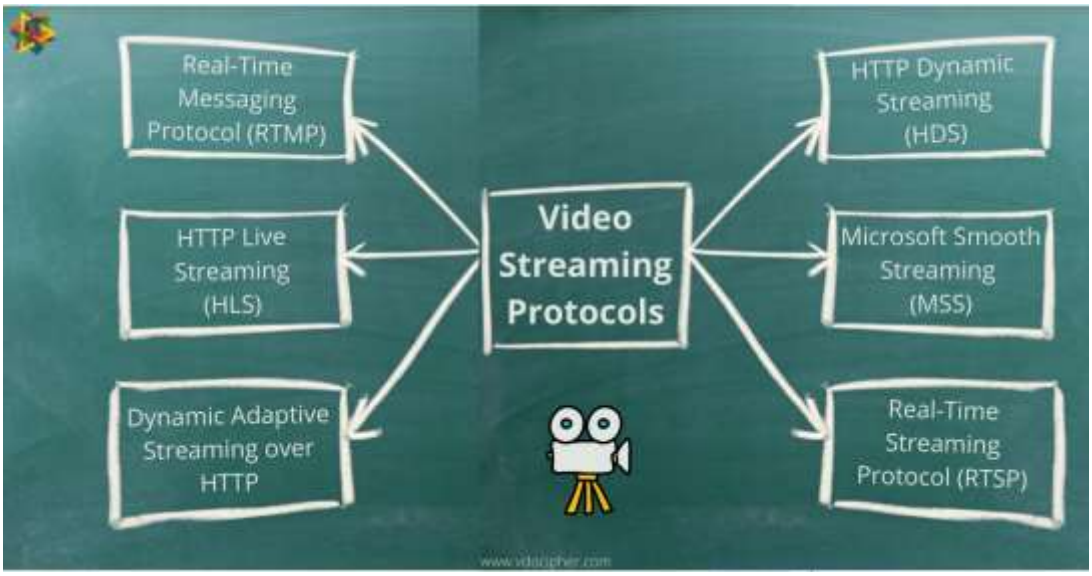
- In my case, this challenge was useful for developing my overall Python programming, particularly in networks. It also extended my skills of critical thinking when designing and launching the network services that correspond to the defined protocols. It has been a quite useful project that has helped me build a clear understanding of DNS, which will be beneficial for more complex networking topics in future work.

Task 5.2D Above and Beyond Credit

Application layer protocols used in video streaming

The application layer protocols facilitate the smooth transmission and reception of video content over the internet. Key protocols include:

- Hypertext Transfer Protocol (HTTP)
- Real-Time Protocol (RTP)
- Real-Time Streaming Protocol (RTSP)
- Dynamic Adaptive Streaming over HTTP (DASH)
- HTTP Live Streaming (HLS)



HTTP

- In my case, this challenge was useful for developing my overall Python programming, particularly in networks. It also extended my skills of critical thinking when designing and launching the network services that correspond to the defined protocols. It has been a quite useful project that has helped me build a clear understanding of DNS, which will be beneficial for more complex networking topics in future work.

RTP

- RTP was developed specifically for real-time data, like audio and video. It uses UDP to lower latency, which is necessary for live streaming: The synchronization of audio and visual streams is facilitated by RTP timestamps. Determining the Payload Type: Facilitates ascertaining the structure of the data being conveyed. QoS, or quality of service, RTP can interact with QoS strategies to maintain streaming quality in the face of varying network conditions.
  RTP's performance can be impacted by network problems like packet loss and jitter, which call for additional protocols for reporting and control.

RTSP

- Systems for communications and entertainment employ a network control protocol called RTSP to control streaming media servers. RTSP is used to create and manage media sessions between endpoints: Commands: Supported VCR-like commands include play, pause, and stop, enabling interactive streaming. The technique of controlling and guiding the material flow during streaming sessions is known as session management.
  RTSP often works alongside RTP to handle control commands during the actual data transport.

DASH

- DASH is an adjustable bitrate streaming system that enables high-quality media streaming over the internet using regular HTTP web servers. Crucial attributes include: Segmented Content: Media files are split up into smaller pieces to allow for adaptive streaming. Manifest File: Also referred to as an MPD (Media Presentation Description), a manifest file instructs the client on which segments to download and play. Adaptive bitrate: Clients can switch between several qualities levels (bitrates) based on the network's condition, providing the optimal viewing experience with the least amount of buffering. Using standard HTTP servers, DASH may easily integrate with existing infrastructure and leverage HTTP-based content delivery.

HLS

- The HLS media streaming protocol, created by Apple, is similar to DASH but has a few key differences: Chunked Transfer: Similar to DASH, HLS divides content into smaller HTTP-based file segments. M3U8 Playlist: During streaming, this list of accessible media segments from an M3U8 file directs the client. Compatibility: Due to its broad support across Apple devices and browsers, it is a popular choice for content creators aiming to reach Apple ecosystems. HLS also provides flexible bitrate streaming, which enhances the user experience across a variety of network conditions.

Protocol Interaction and Integration

- In reality, video streaming typically makes use of a combination of these protocols: RTSP and RTP are often used for real-time interactive applications such as live sports broadcasting. DASH and HLS are commonly used for Video on Demand (VoD) services due to their adaptive streaming capabilities and reliance on HTTP, which allows them to benefit from CDNs. Through the use of HTTP/HTTPS, which reduces latency and speeds up load times, content delivery networks (CDNs) may more efficiently and locally distribute media assets.

Emerging Trends and Protocols

- Video streaming methods are continually evolving due to advancements.
  Peer-to-peer streaming between browsers is made possible via WebRTC, or Web Real-Time Communication, which is utilized in applications like video conferencing and does away with the need for intermediate servers.
  Google introduced QUIC (Quick UDP Internet Connections), which minimizes connection establishment time and boosts congestion control to address TCP's limitations, notably for streaming.

- A deep comprehension of the application layer protocols for video streaming exposes a complex ecosystem in which multiple protocols frequently work together to offer smooth, high-quality video content. This exchange demonstrates the vital role that application layer protocols play in the modern digital media ecosystem by guaranteeing efficient, adaptable, and dependable video delivery across a variety of devices and network conditions.

Active class 7: Who is Instructing (Module 5)

3 Widely Used Routing Protocols

- ✓ Routing Information Protocol (RIP)
- ✓ Open Shortest Path First (OSPF)
- ✓ Border Gateway Protocol (BGP)

Feature	RIP	OSPF	BGP
Protocol Type	Distance-vector	Link-state	Path-vector
Metric	Hop Count	Cost (bandwidth delay)	Path attributes
Algorithm	Bellman-Ford	Dijkstra	Path vector algorithms
Max Hop Count	15	Unlimited	Not applicable
Convergence Speed	Slow	Fast	Slow
Scalability	Low	High	Very high
Resource Usage	Low	Moderate to High	Moderate to High
Configuration	Simple	Complex	Very Complex
Use case	Small networks	Large enterprise networks	Internet and large-scale inter-AS routing

A description of every protocol

RIP

- RIP helps computers inside a network choose the most efficient means of data exchange, much like a simple messenger. It counts the number of "hops" or steps that separate two computers. Think of hops as rest stops along the way. A message will not be allowed to pass via RIP if it must pass through more than 15 checkpoints.
  - ✓ Because RIP is simple to understand and configure, it works well for smaller, more basic networks. Owing to its little resource usage, it functions effectively in environments with constrained memory and processing capacity.
  - ✓ One of the main disadvantages of RIP is that it can only support networks up to a maximum of 15 hop counts, which limits the size of networks it can support. Furthermore, its slow convergence time may lead to short routing loops and wasteful routing.

OSPF

- OSPF is similar to a more sophisticated messenger that covers the whole network, much like GPS maps roads. It understands the quality of those steps as well as their overall number (or hops), accounting for factors like speed limitations on public roads. It uses a method called Dijkstra's algorithm to figure out the fastest and most economical path for messages to take.

    - ✓ OSPF is highly scalable and provides rapid convergence since it is link-state based and employs the Dijkstra algorithm. It supports multiple indicators to choose the best way, making routing decisions more reliable and efficient. OSPF also makes hierarchical network design easier using the concept of regions, which enhances scalability and management.
    - ✓ Keeping track of the complete network topology might cause problems with resource usage and complicate the OSPF configuration. It is less suitable for small, simple networks than RIP since it requires more CPU power and memory.

BGP

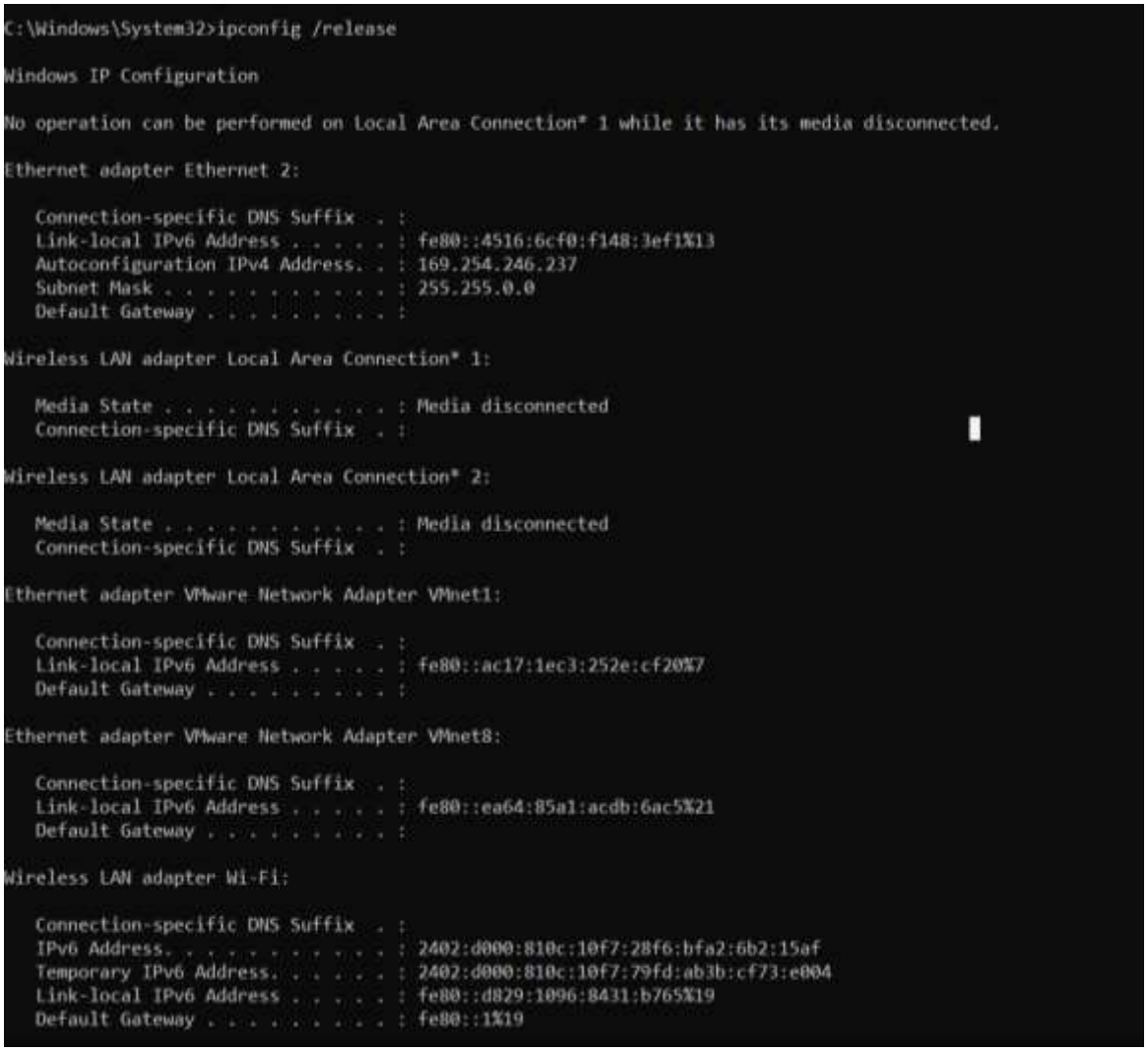
- BGP is similar to the international travel guide on the internet. It makes it easier for numerous large networks—often referred to as autonomous systems, or ASes—to communicate and share routes. In addition to hops and speed, BGP takes into account other variables like customs laws, tolls, and airline connections to determine the best route.

    - ✓ BGP is essential for internet routing because it handles enormous volumes of routing data between multiple autonomous systems. Its powerful policy-based routing capabilities allow administrators to design complex routing policies based on a range of parameters. BGP, the backbone of the internet, is unmatched in terms of scalability and endurance.
    - ✓ Because of its intricate setup and functioning, BGP demands a high level of expertise.
      Its slower convergence time when compared to OSPF could be a concern when rapid network modifications are required.

Active class 8: Internet is full of Network Protocols (Module 5)

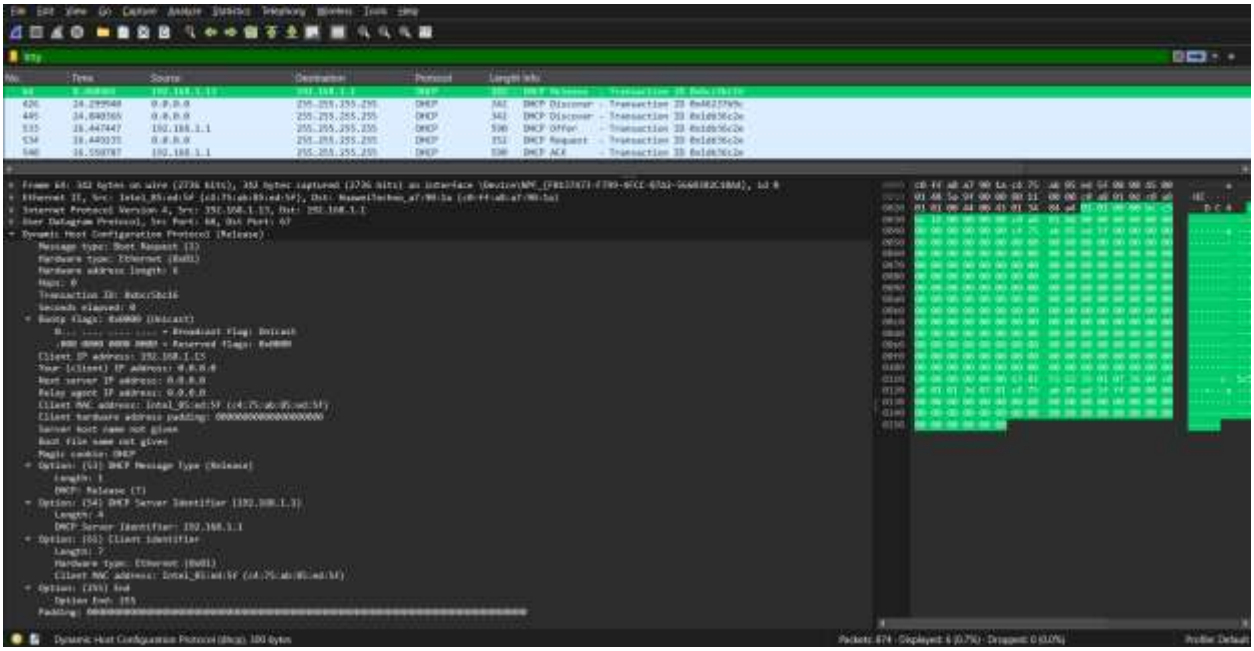
How to Use Wireshark to Capture DHCP Packets

- Start Wireshark: Turn on Wireshark on my computer
- Choose the Network Interface: I selected the network interface (such as an Ethernet adapter or Wi-Fi network) on which I wanted to record packets.
- Start Capturing: Next, I selected the shark fin-shaped button labeled "Start Capturing."
- Command Line: Next, I typed the commands ipconfig /release to release my computer's IP addresses, ipconfig /renew to obtain new IP addresses, and ipconfig /all to double-check.



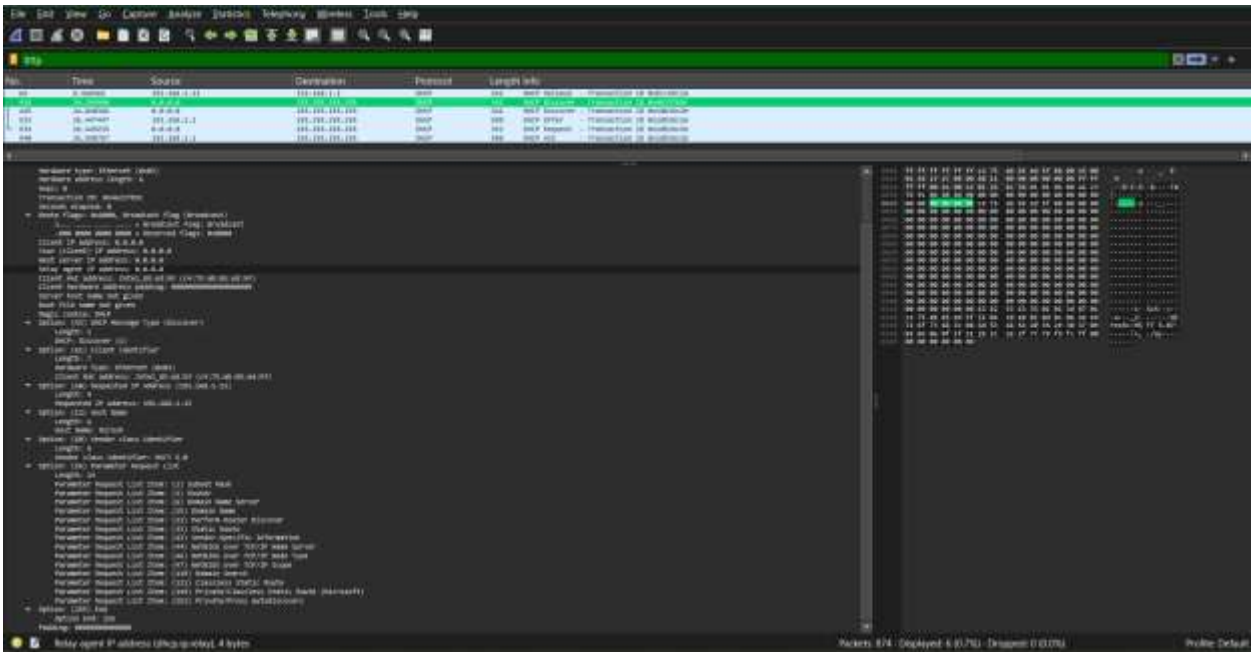
Returning to Wireshark I now halt the capture and enter the dhcp filter to discover the subsequent DHCP message sequence.

DHCP Release



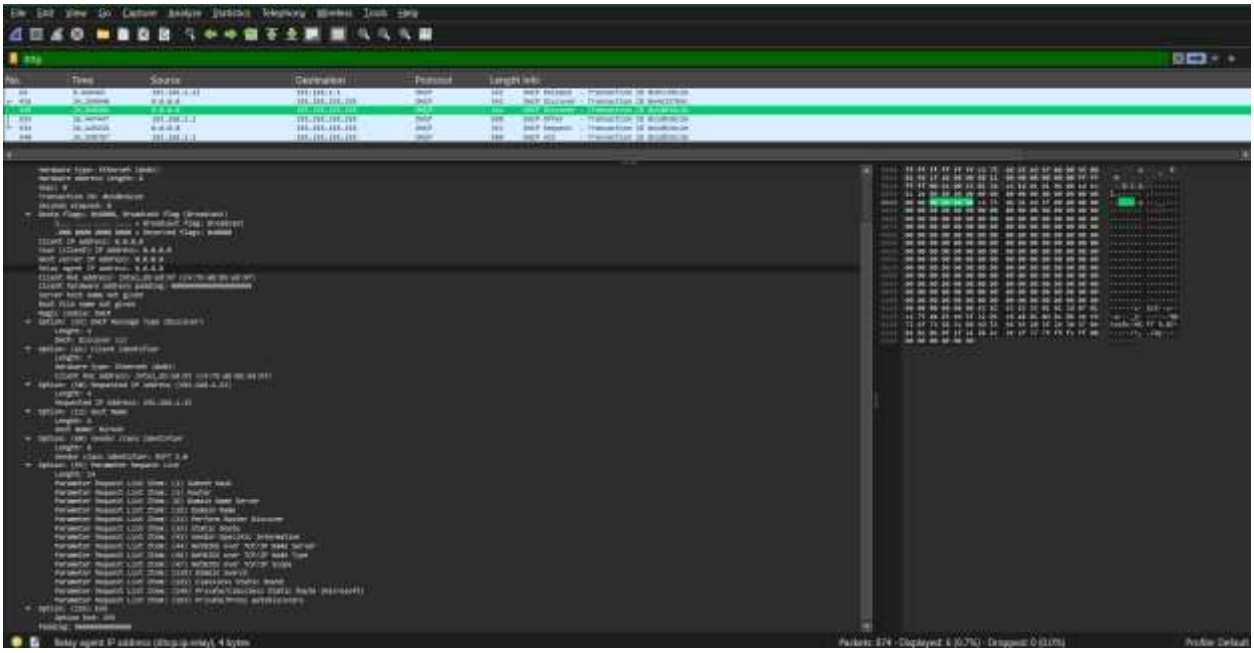
DHCP Discover

- To join a network and get an IP address, a device needs to transmit and receive a series of DHCP (Dynamic Host Configuration Protocol) signals. The process is started with the DHCP Discover message. Here, the device broadcasts a message to find any DHCP servers that can provide it an IP address. This message has no specified destination because the device does not yet have an IP address.



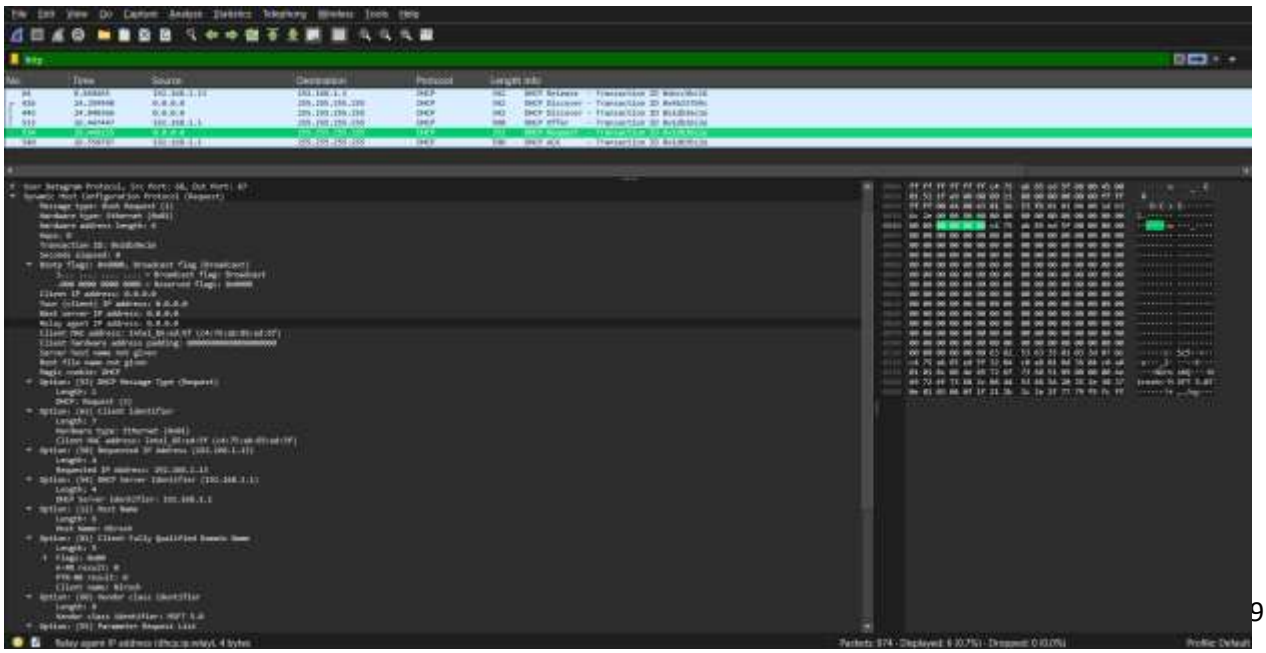
DHCP Offer

- The DHCP Offer message is subsequently sent by a DHCP server that has received the Discover message. The server responds with the device's IP address and some additional information, such as the subnet mask and the lease term (the duration for which the device is allowed to use this IP address).



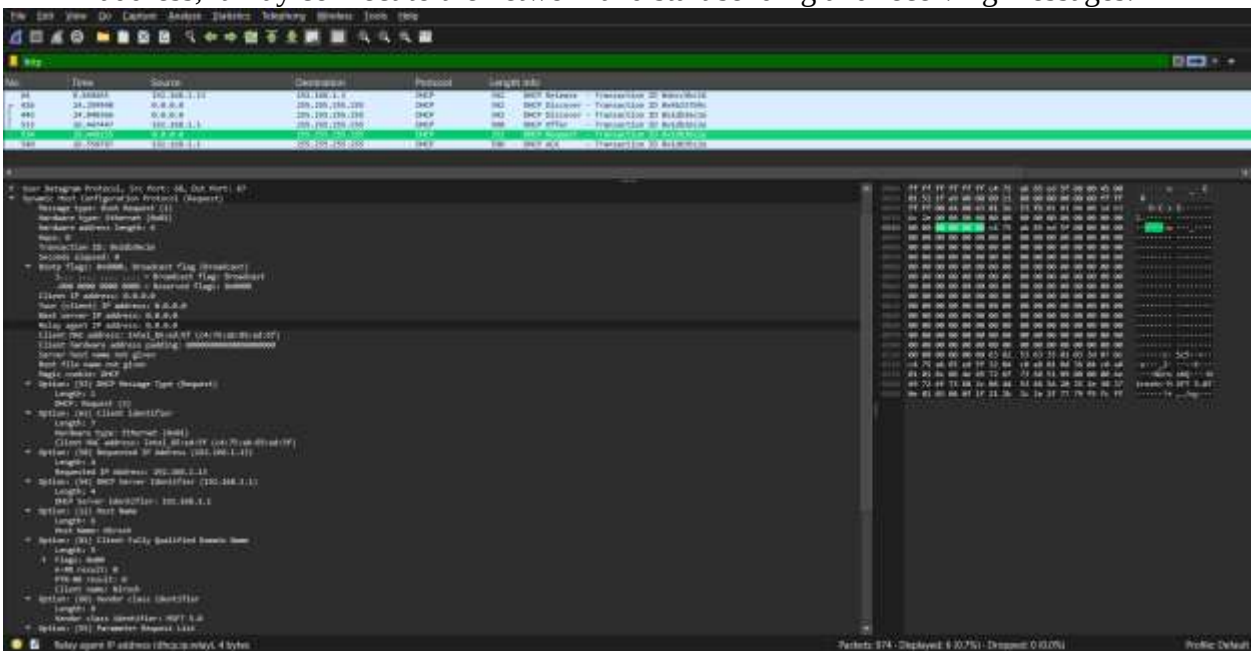
DHCP Request

- The device then responds with a DHCP Request message. This message is also broadcast so that it is seen by all DHCP servers. The device is informing other servers that it is accepting an offer from a particular server and is asking for the IP address that server is providing, based on the Request message.



DHCP Acknowledgment (ACK):

- Lastly, the DHCP server verifies the lease by sending out a DHCP Acknowledgment (ACK) message. Finally, this message includes the IP address, subnet mask, default gateway, and lease length as assignment parameters. Now that the device has the IP address, it may connect to the network and start sending and receiving messages.

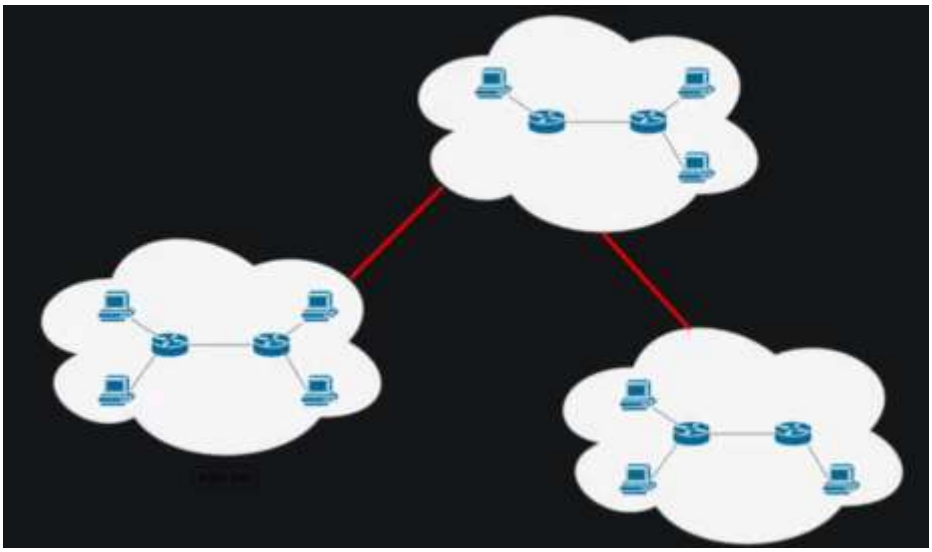


- In summary, the device sends out a Discover message to find a server; the server replies with an Offer; the device requests the IP address that is supplied; and the server confirms by sending out an ACK. Through this procedure, the device's IP address and configuration are guaranteed to be correct when it joins to the network.

Inter-AS Protocols

- ✓ Traffic between several autonomous systems (ASes), which are sizable networks or clusters of networks under a single management, is routed via these protocols. They function between separate networks, frequently with various management and policy setups. They also offer tools for intricate agreements and routing policies amongst autonomous systems.

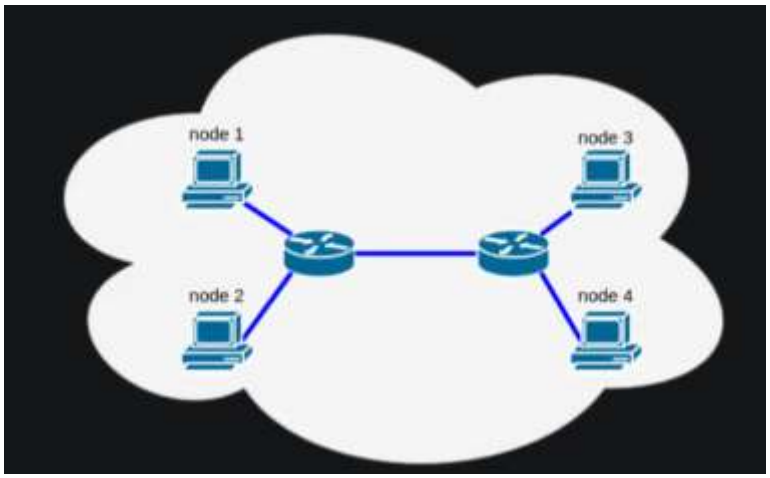
Eg : order Gateway Protocol (BGP)



Intra-AS Protocols

- ✓ Within a single autonomous system, traffic is routed via these protocols. Under a single administrative domain, they function as a single, unified network. They prioritize criteria like shortest path, bandwidth, and delay in order to concentrate on effective routing within the AS.

Eg: Open Shortest Path First (OSPF)
Enhanced Interior Gateway Routing Protocol (EIGRP)



- Inter-AS protocols like BGP are used to handle large-scale, autonomous network routing. They also offer sophisticated policy control. Effective routing within a single network is the focus of intra-AS protocols like OSPF and EIGRP, which use metrics to ensure the best possible path selection. Network administrators can more effectively design and manage internal network operations and large-scale internet routing by having a thorough understanding of these protocols.

Summary and Reflection for Above and Beyond Tasks in Class 7 and 8

Summary

- To summarize, the advanced topics pertaining to network protocols and routing were covered in the above-and-beyond assignments in Classes 7 and 8. In Class 7, we looked into modern routing algorithms and discussed the advantages and disadvantages of both static and dynamic routing protocols. In class, we recorded and examined the series of DHCP communications that are exchanged when IP addresses are assigned, using Wireshark to study DHCP. Additionally, we talked about the distinctions and applications of the two kinds of routing protocols—intra-AS and inter-AS (Autonomous System)—providing examples of both.

Reflection

- Through these exercises, I was able to fully understand complex networking concepts. In Class 7, the benefits of a comparison of the two kinds of routing protocols highlighted the advantages of automated routing decisions in dynamic protocols, such as greater scalability and fewer manual setting. The advantages of static routing in small networks were realized at the same time. It is evident from looking at current routing algorithms how important efficient routing methods are to preserving network dependability and performance.

- In Class 8, I gained a grasp of the dynamic IP allocation process and learned how to use Wireshark to analyze DHCP packets. This helped me solve issues with network connectivity. Understanding the differences between intra-AS and inter-AS routing protocols, which highlight the need of using different tactics to handle internal and external network traffic, has broadened my perspective on routing in large-scale networks. These advanced exercises have improved my understanding of network administration and diagnostics by highlighting the critical roles that efficient routing and IP address management play in the reliability and performance of networks.

Active class 9: Data-link Layer (Module 6)

Security issues associated with ARP

- The Address Resolution Protocol (ARP) maps IP addresses to MAC addresses on a local network. Although ARP is required for network communication, there are several security holes in it that attackers might exploit. One of ARP's primary issues is the absence of any internal security measures to verify the authenticity of ARP messages. It is hence vulnerable to several types of attacks.

ARP Poisoning (ARP Spoofing)

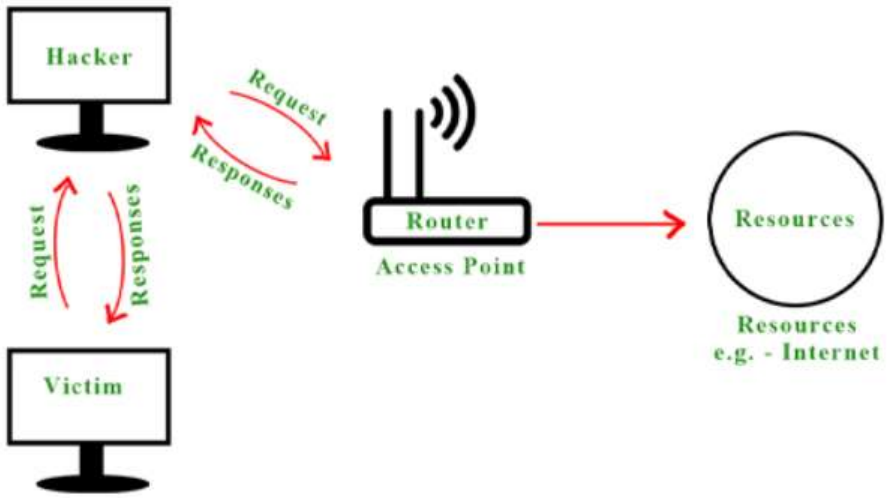
- By sending phony ARP messages, an attacker can fool the network through ARP spoofing. These connections connect the MAC address of the attacker to the IP address of an authorized device (such a gateway or another computer). With this knowledge, the attacker can then intercept, modify, or halt data intended for the approved device. Ultimately, the attacker tricks the network into transmitting data to their device instead of the correct one.

Static ARP Entries

Man-in-the-Middle Attack

- ARP spoofing may lead to a man-in-the-middle (MITM) attack. In this instance, the assailant inserts themselves covertly between two communication devices. This allows the attacker to potentially intercept and alter the data being sent. For instance, confidential information like passwords or financial information may be stolen during a Man-in-the-Middle attack.

Encryption



Denial of Service (DoS) Attack

- Attackers can also use ARP spoofing to perform DoS assaults. By sending a large number of bogus ARP packets, the attacker can overwhelm the network and cause legitimate devices to receive erroneous ARP information. This could impede normal communication, slow down the network, or render it unusable for authorized users.

ARP Inspection

Dynamic ARP inspection is a feature of some sophisticated network switches that verifies ARP packets prior to processing.

Summary and Reflection for Above and Beyond Tasks in Class 9

Summary

- The class's above-and-beyond goals centered on identifying and understanding Address Resolution Protocol (ARP) security vulnerabilities. We examined and discussed how ARP attacks, such as ARP spoofing and ARP poisoning, affect network security. We also investigated potential mitigation strategies to protect networks from these security vulnerabilities.

Reflection

- The security flaws pertaining to the data link layer were mostly brought to light by this assignment. I was able to identify the various ARP attack types and have a better understanding of how attackers can utilize ARP vulnerabilities to intercept or alter network communication. To safeguard networks against these kinds of intrusions, the application of robust security mechanisms, including ARP inspection, dynamic ARP inspection, and static ARP entries The topic of spoofing detection technologies was brought up during the mitigation alternatives discussion.

- This exercise highlighted the need of being proactive and on guard when it comes to network security, particularly at the data connection layer where even seemingly simple protocols can become the subject of sophisticated attacks. It emphasized how important it is for cybersecurity experts to continuously learn and adjust in order to successfully protect network infrastructure from evolving threats.

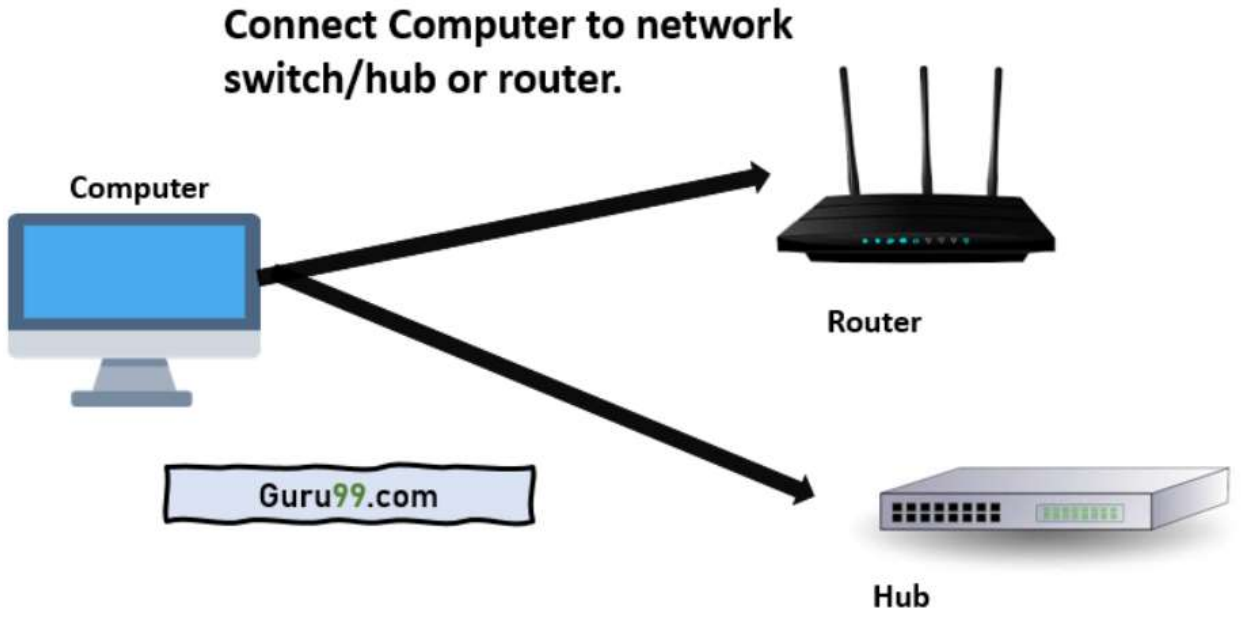
Active class 10: Physical Connections and Protocols (Module 7)

What is an Ethernet cable

- An Ethernet cable allows two devices to be connected to a wired network at a high speed. This network connection is composed of four pairs of twisted pair transistors. At both ends of the line, data transfer is accomplished via the RJ45 connector. He many kinds of Ethernet cables are Cat 5, Cat 5e, Cat 6, and UTP cables. Cat 5 cable can only handle networks operating at 10/100 Mbps, but Cat 5e and Cat 6 cables can support Ethernet networks operating at 10/100/1000 Mbps.

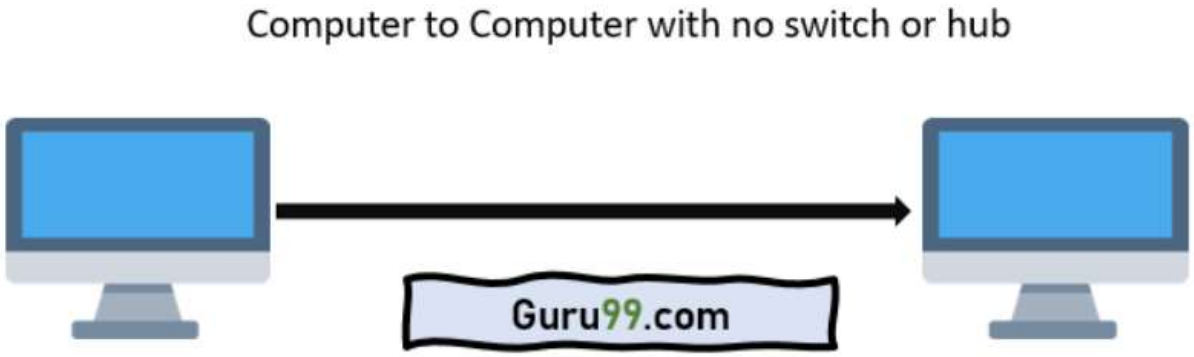
Straight through cables

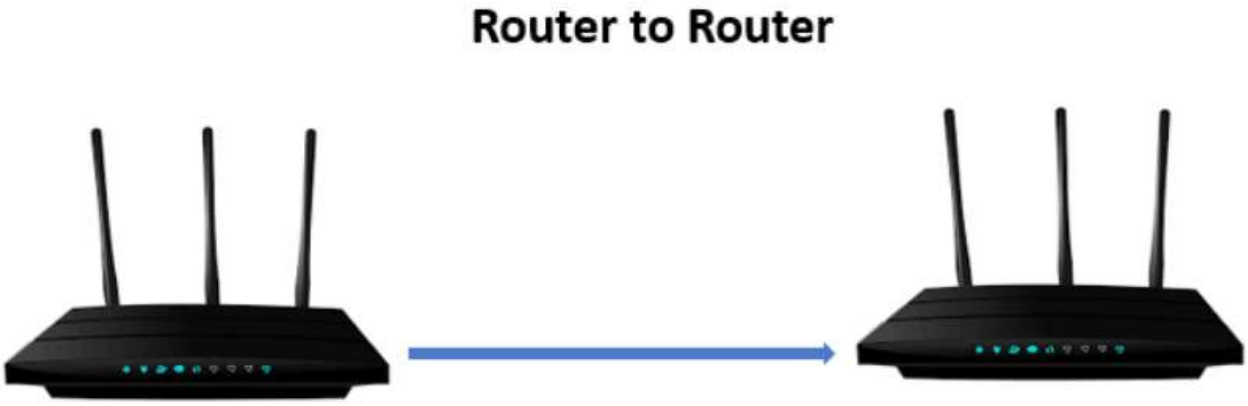
- A straight through Ethernet cable is the most often used type. Because the cable has the same wiring on both ends, the wires within are connected in the same order at both ends. Straight through cables are used to connect different devices to one another, such as a computer to a switch or router or a switch to a router. For instance, if a desktop computer were linked to a router, it would be connected to the internet using a straight through cable.



Crossover cables

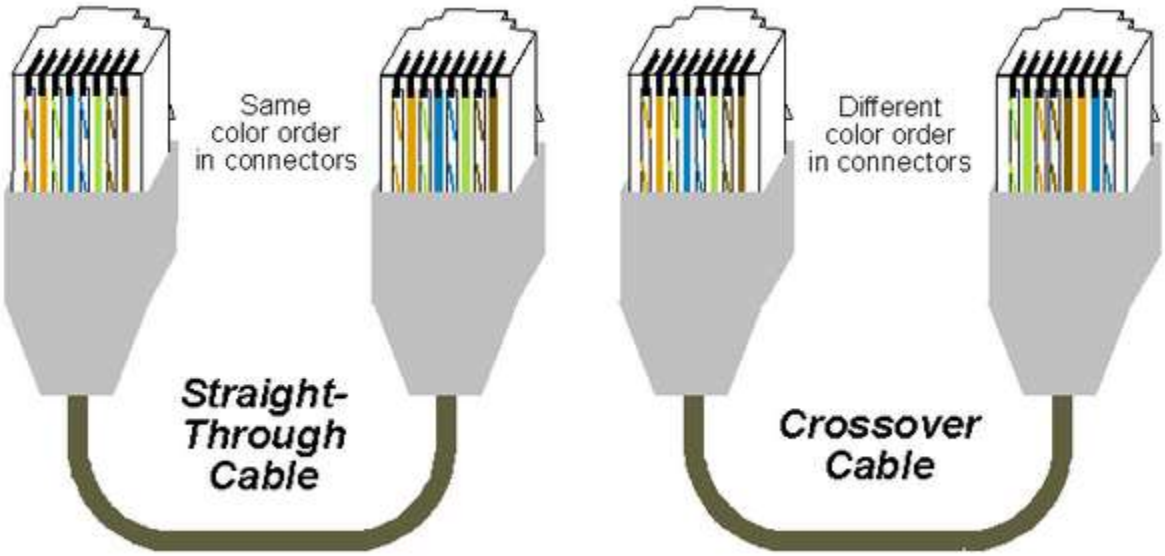
- A crossover Ethernet cable features distinct wiring on each ends, with the wires crossed, giving the cable a distinctive wire sequence at each end. Without the use of a router in between, crossover cables are used to connect comparable types of equipment directly to one another, such as a switch to another switch or a computer to another computer.
  If you wanted to transfer files directly between two computers without using the internet, for example, you would use a crossover connection.





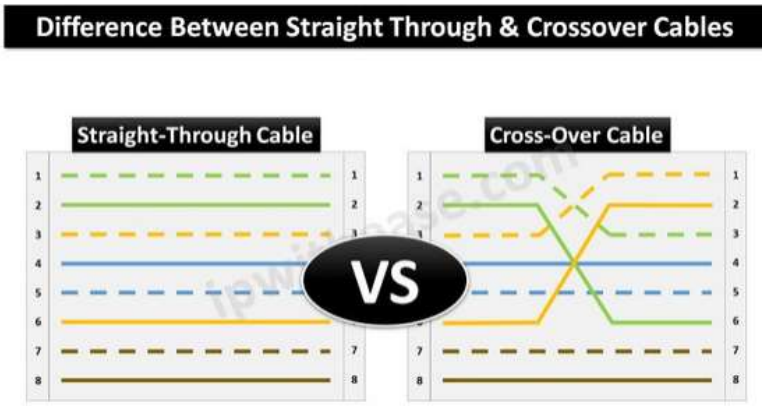
How to identify these 2 cables

- An Ethernet cable's ends can be used to identify if it is a straight through or crossover cable. Holding the two ends of the cable side by side, examine the colored wires inside the connectors (RJ45 connectors). In a straight through wire, the colors are arranged the same on both ends. A crossover cable's colors will appear in a different order on either end.



Straight Through Cable: 1-1, 2-2, 3-3, 4-4, 5-5, 6-6, 7-7, and 8-8 are the pin configurations.

Crossover Cable: Depending on the particular wiring standard being used, the remaining pins may remain in their original positions or may also be crossed. The pins that are most crucial for crossing are arranged as 1-3, 2-6, 3-1, and 6-2.



Summary and Reflection for class 10

Summary

- I looked into the two main kinds of Ethernet cables—straight through and crossover cables—during this assignment. The connections that make up computer networks. I began by outlining the uses for a straight through Ethernet line. This type of cable is typically used to connect multiple devices, such as a switch or router and a computer. Next, I discussed the use of crossover Ethernet cables and their definition. When establishing comparable A crossover cable is used to connect two pieces of equipment, such as switches or a computer, directly to one another. Lastly, I described how to look at the wiring sequence on both ends to identify the type of Ethernet cable. Straight through cables have the same wiring pattern on both ends, whereas crossover cables have distinct designs, as I mentioned.

Reflection

- It's essential to understand the differences between crossover and straight through Ethernet connections in order to configure and troubleshoot networks properly. I now realize how vital it is to choose the correct kind of cable to assure efficient network device communication thanks to this activity. By understanding when to use each type of cable, network efficiency is boosted and frequent connectivity issues are avoided. Furthermore, I gained a crucial skill for configuring new networks and repairing old ones when I mastered the ability to identify wires based on their wiring patterns. Since it sets the framework for more complicated networking concepts and procedures, this understanding is crucial for anyone working in network administration

INTERNET OF THINGS (IOT)

KENISHA CORERA|DFCS|DK|62|203| SIT 202

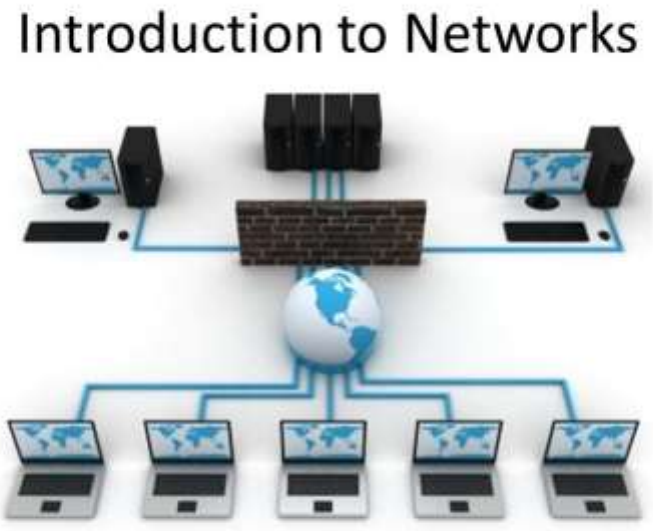
TABLE OF CONTENT

Introduction

- Computer networks are undergoing a fundamental change in their context. Three principal themes motivate this change from static, traditional network to an interconnected world: network security, internet of things (IoT)-imagine a future where smart cities are indeed like computer networks! Instead of being inflexible and unvarying, they are as variable in their responses to program changes as traffic lights reacting to current road traffic.

- The Internet of Things, or IoT, is like having all of the city's shops, cars, and homes networked and talking to each other! It's a massive network of devices that includes fitness trackers and smart appliances.

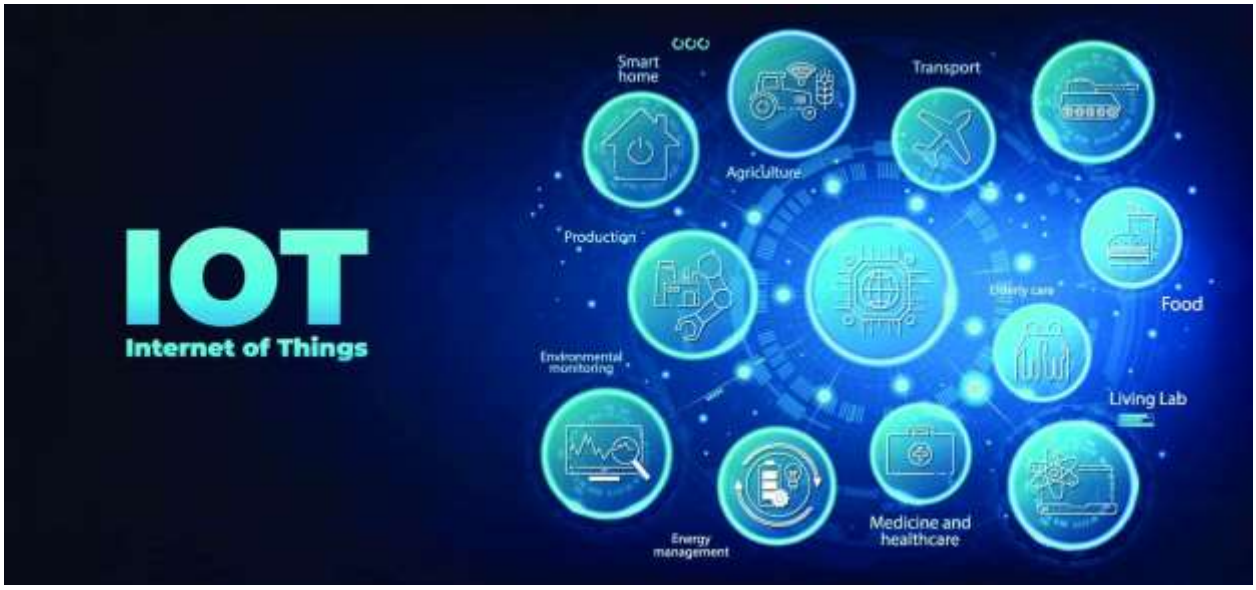
- This paper delves deeper into these themes, elucidating how they're transforming network architecture and management to make networks more adaptable and safe for the future.



Internet of Thing (IOT)

What is ITO?

- ✓ The Internet of Things (IoT) is a network of actual objects, or "things," that are outfitted with sensors, software, and connections to enable them to collect and exchange data over the internet [14].



How IoT differs from traditional networking?

- Traditional networking focused on connecting computers and servers, while the Internet of Things (IoT) broadens connectivity to ordinary objects, enabling data interchange and remote monitoring and control. [15]

Motivation behind IoT

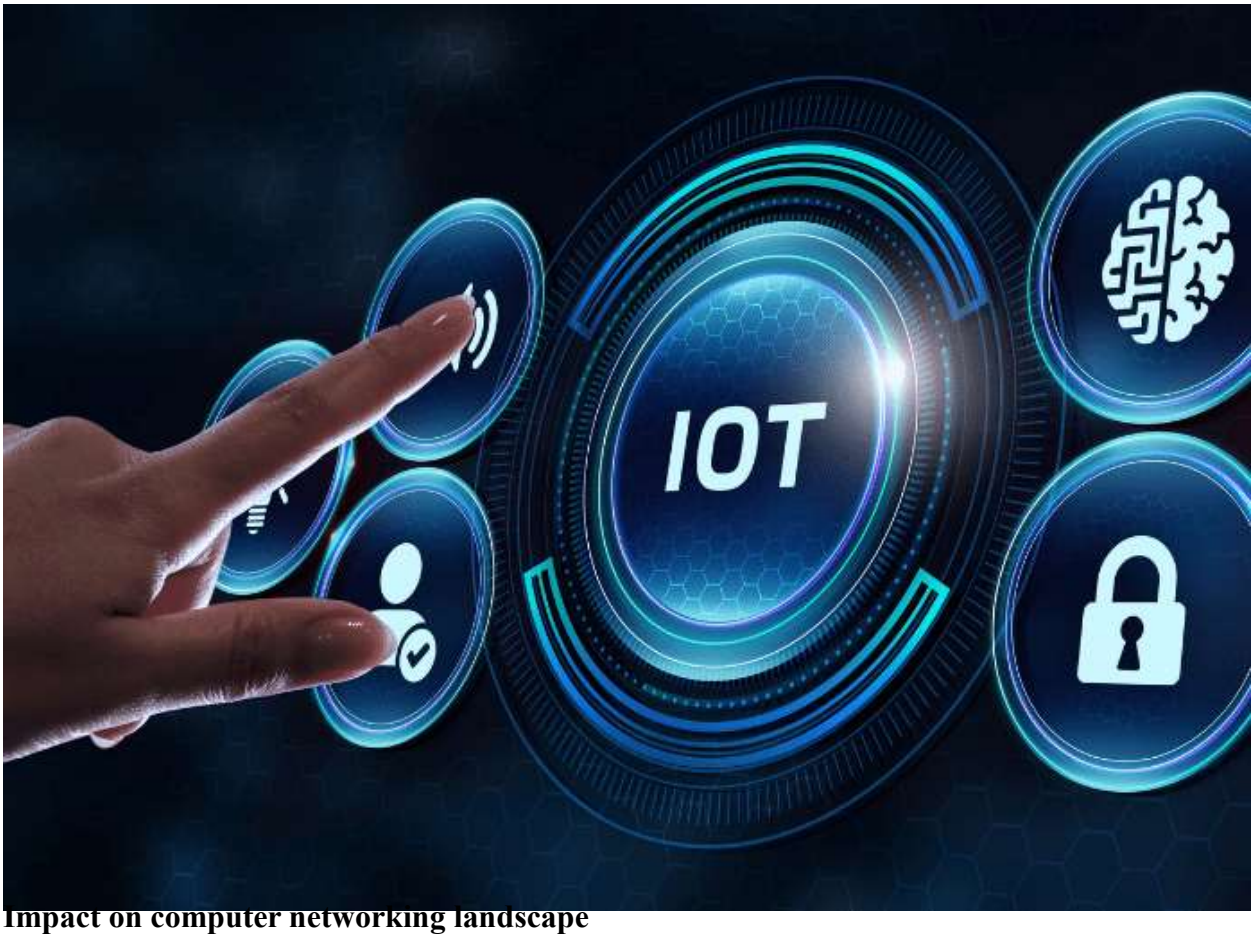
- IoT is mainly motivated by the desire to facilitate seamless communication and integration between various platforms and devices, which will increase output, automate procedures, and improve data-driven decision-making. [16]

Key technologies and concepts

- **Sensors and actuators**: Internet of Things devices employ sensors to collect data and actuators to perform actions.
- **Wireless communication methods:** Devices in the Internet of Things employ cellular networks, Bluetooth, Wi-Fi, ZigBee, and other wireless protocols to communicate.
- **Cloud computing and data analytics:** Cloud processing and analysis of IoT data is common for decision-making and insights. [17]

Applications of IoT

- The Internet of Things is used in a wide range of sectors and fields, including smart cities, smart homes, smart manufacturing, smart transportation, smart agriculture, and smart cities [18].



Impact on computer networking landscape

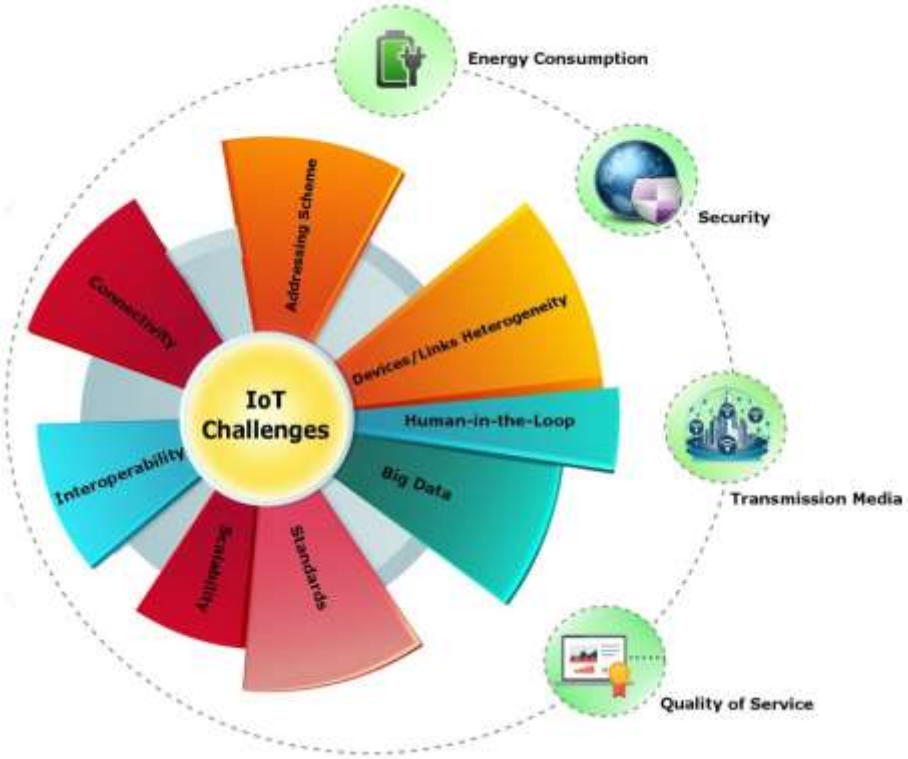
- The Internet of Things is driving the need for more dependable, scalable, and secure computer networks in order to manage the massive volume of data generated and the huge number of connected devices [19].

Changes required for current networking paradigm

- To allow IoT, computer networks must adapt to handle the increased traffic, provide better scalability and security, and implement enhanced security measures to thwart any attacks. [20]

What are the challenges in IoT?

- Because linked devices are vulnerable to hacking and data breaches, security and privacy pose the biggest problems in the Internet of Things. Another difficulty is interoperability because different devices frequently employ different communication standards and protocols. As the quantity of IoT devices increases and strains network infrastructure, scalability becomes a challenge. Another issue is power management, particularly with battery-powered gadgets. Massive data generation also increases the complexity of data management and storage. The utility of IoT devices might be limited by network dependability in some circumstances, such as distant places, and privacy standards require regulatory compliance.

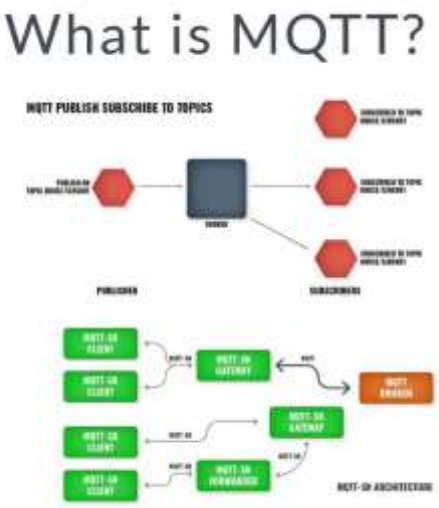


What are the different protocols used to support IoT use cases? Discuss different application protocols used.

- IoT uses a variety of protocols to accommodate various use cases. Web-based IoT devices frequently employ HTTP (HyperText Transfer Protocol), yet this protocol might not be appropriate for low-power or restricted situations. For devices with limited resources, the lightweight CoAP (Constrained Application Protocol) protocol is appropriate. Advanced Message Queuing Protocol, or AMQP, provides dependable communication in dispersed settings. MQTT (Message Queuing Telemetry Transport) is developed for low-bandwidth, high-latency networks. While Bluetooth and Wi-Fi provide device-to-device connection, Zigbee and Z-Wave are also utilised in smart home applications for low-power, short-range communication.

What is MQTT?

- MQTT (Message Queuing Telemetry Transport) is a lightweight messaging protocol that is perfect for Internet of Things applications because it is made for devices with limited bandwidth, high latency, or unstable networks. It makes use of a publish-subscribe paradigm in which clients, or devices, post messages to particular subjects, and subscribers, or other people, receive these messages. The protocol, which uses TCP/IP to function, is made to use the least amount of device resources and network traffic possible. Moreover, it provides Quality of Service (QoS) tiers to guarantee dependable message delivery. Because of its simplicity and effectiveness, MQTT is frequently utilised in Internet of Things systems for applications like home automation and remote monitoring.



How do I Implement a smart home that includes several smart appliances (an IoT use case) in Cisco packet tracer?

- Start by adding smart appliances from the IoT device library, such as security cameras, smart lights, and thermostats, to create a smart home with IoT devices in Cisco Packet Tracer. Make wired or wireless connections to a home gateway (router) from these devices. Make that the gateway is linked to the internet or a local network. To enable control, configure the appliances using the IoT Registration Server. By implementing an Internet of Things protocol such as MQTT, you can emulate device control from a laptop or smartphone. Lastly, use remote device control to test the configuration by checking camera feeds, changing temperature settings, and turning on or off lights.

How do I analyse MQTT protocol in Cisco packet tracer (tip: you can implement a MQTT broker in laptop, mobile connected to the LAN)? The following figure shows a sample smart home implementation in Cisco Packet Tracer. The analysis should include 1–2-minute video capture of the cisco packet tracer screen of your own implementation and MQTT in action (with your audio explanation).

https://drive.google.com/file/d/1sw4ZJ0d6BrEYcAM63zKaZF8O3aA1eto0/view?usp=drive_link

Conclusion

By incorporating commonplace items into networked ecosystems, the Internet of Things (IoT) is revolutionising established networking infrastructures and marking a major advancement in networking. The Internet of Things (IoT) facilitates the smooth exchange of data between devices by utilising essential technologies such as sensors, wireless communications, and cloud computing. This allows for real-time insights and automation. As we've seen, this transition brings with it both opportunities and difficulties, including the requirement for improved network interoperability, security, and scalability. Furthermore, protocols like AMQP, CoAP, and MQTT are essential for facilitating effective communication in Internet of Things contexts, particularly for devices with limited resources.

IoT has enormous promise; its uses span from smart cities and homes to healthcare and agriculture. To fully benefit from it, though, existing network concepts must change to meet the increasing traffic and security needs of billions of linked devices. IoT is positioned to transform daily life and industries as it develops further, pushing the limits of automation and connectivity.

Reference

- **Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M.** (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7), 1645-1660.
- **Mouradian, C., Naboulsi, D., et al.** (2018). A comprehensive survey on fog computing: State-of-the-art and research challenges. *IEEE Communications Surveys & Tutorials*, 20(1), 416-464.
- **Höller, J., Tsiatsis, V., Mulligan, C., et al.** (2014). *From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence*. Academic Press.