CSN-341
Computer Networks
Assignment - 2

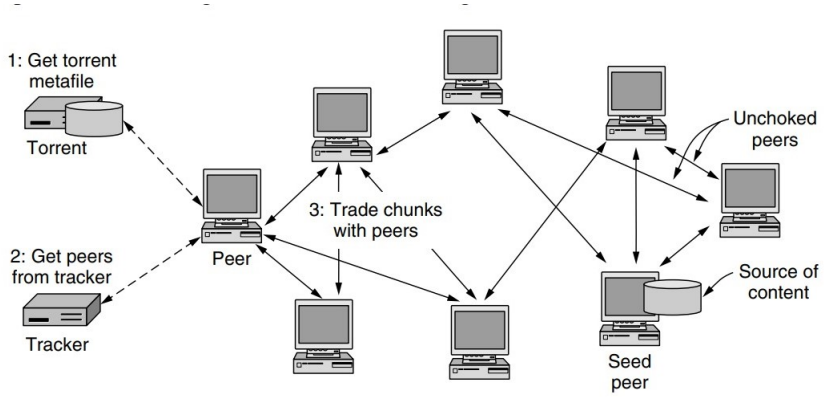
Group-6

Name	Enrollment No
Meet Sindhav	22114053
Mohammed Haaziq	22114055
Aditya Mundada	22114058
Nayan Kakade	22114060
Sarvesh Baheti	22114087
Roopam Taneja	22125030

Question 1: Explore the mechanisms through which a new peer like Alice in a BitTorrent network acquires her first chunk. Discuss the fairness and efficiency of these mechanisms, and consider how different scenarios (e.g., network congestion, peer availability) might affect Alice's experience.

BitTorrent network is a **P2P protocol** designed for sharing large files among a set of peers.
It can be of two types:-

1. BitTorrent with Tracker
   - Tracker, is the centralized server that coordinates the file sharing process.
   - When a new peer joins the network, it asks the tracker for a torrent file. This file contains metadata of the information of the chunks in the content file and its address.
   - The new peer sends a request to any of the neighboring peers that contains any of the chunks.
   - The neighboring peers respond with the data and in this way Alice acquires her first chunk.
2. BitTorrent without Tracker
   - This network is a distributed network where not all nodes are connected to each other.
   - We use Distributed Hash Table (DHT) to look for the peers that have the chunk.
   - In the table we use a key (which is the torrent description) and the value stored at the key contains the peers that have the required data.



Fairness:-

- The Tit-for-Tat strategy used by BitTorrent describes the fairness of the mechanism. After you get your first chunk, it is important to help other peers get the chunk you have. The more you participate in uploading data, the more fruitful it is for you to download the remaining data.
- The seeders are also kind enough to share the first chunk to the peers having no data so that they can participate in the network through the optimistic unchoking process.

Efficiency:-

- As files are divided into chunks, the peers download the chunks from different peers. The parallelism speeds up the overall process.
- It also optimizes the use of available resources (example - bandwidth)
- Due to the Tit-for-Tat mechanism, each peer involves in sharing the data.
- Due to the Rarest first mechanism, the rarest chunk is downloaded first so that it is made available to the other peers. This prevents the network from creating a bottleneck for a single chunk.
- Due to data redundancy, even if one peer fails, other peers still have the data.
- It also eliminates the central server failure problem since the network uses the Distributed Hash Table (DHT) mechanism.

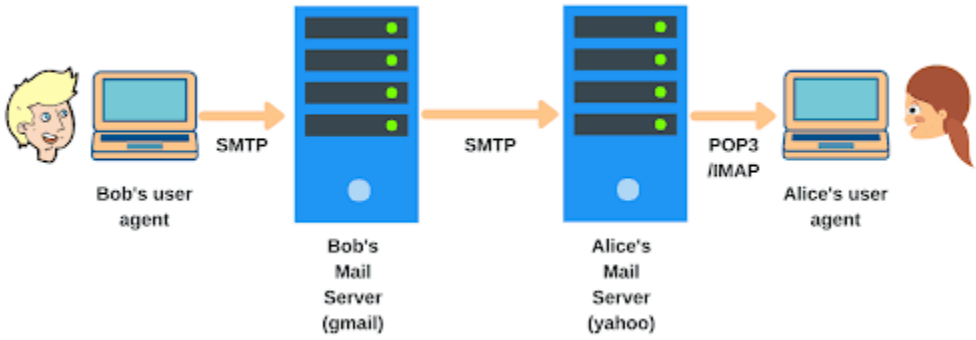
Network Congestion:-

- Due to network congestion, the download speed will reduce and latency will increase.
- Connecting to other peers will become time consuming.
- Due to overload, packet loss can take place which will lead to repetitive asking for the chunk leading to more congestion.
- This can also lead to breaking of connection between peers.

Peer Availability:-

- If the seeders (nodes which have the complete data) are less in number, it can lead to increase in latency of file sharing.
- If the leechers don't collaborate in uploading the data they have, then it will lead to an increase in latency and resulting in an unbalanced network.
- If after acquiring all chunks, the peer leaves the network, then it will break the network which can lead to an increase in latency for file sharing.

Question 2: Evaluate the challenges and limitations of using POP3 for email access across multiple devices, particularly with the "download and keep" strategy. Propose potential solutions or alternative protocols that could mitigate these challenges, and justify your recommendations.



POP3 (Post Office Protocol v3) is a message access protocol. It allows the receiver to access mail received in its mail server (that is the third step in the above diagram). SMTP (Simple Mail Transfer Protocol) cannot be used for the purpose since it is a push protocol, while obtaining messages is a pull operation from the server to the client.

POP3 is a simple protocol but has its limitations. In POP3, users can download his/her messages via a TCP connection to port 110. POP3 provides "download and delete" or "download and keep" services. The former removes the messages from the server once they are downloaded while the latter makes the server retain them. Although it was widely used earlier, its popularity has been on a decline due to its several limitations.

Almost everyone today uses multiple devices in their day-to-day life. However the limitations of POP3 severely impact the ability to access mail from multiple devices. They include :

- Allows mail to be accessed only by a single device at a time. Restricts users from using multiple devices.
- Does not synchronize changes from one device (like read/unread status) to the mail server. This causes inconsistency and confusion.
- If opted for "deleting" mail from the server upon downloading, the mail is then available only in the memory of the local device. This makes it inaccessible to other devices and also vulnerable to data loss in case of any damage to the device.
- Need to download the mail to read it. This is in contrast to IMAP4 (Internet Mail Access Protocol v.4) which allows the email header and a portion to be read without downloading. This is a problem in areas with low bandwidth.
- Does not allow creating folders on the mail server. This makes organizing mail tougher.

Potential solutions and alternative protocols :

1. **IMAP4** (Internet Mail Access Protocol v.4) :
   IMAP4 is a more powerful protocol than POP3. It allows :
   - Accessing mail from multiple devices simultaneously.
   - Synchronises changes to the mail server and maintains consistency.
   - Allows to read the mail partially without downloading.
   - Allows to create a folder hierarchy for better organizing mail.

Justification: IMAP seamlessly allows multiple devices to access mails and hence is the most popular message access protocol. This makes it ideal for users who need to access their email from various locations and devices, ensuring consistency and synchronization.

2. **Implement email backup solution** :
   Although not recommended, if the user wishes to continue with POP3, he/she is advised to implement a robust backup mechanism (like a cloud-backed solution) to avoid the risk of losing mail.

   Justification: Regular backups ensure that mails are not lost in case of device failure or accidental deletion. However, it doesn't solve the issue of multi-device synchronization.

Question 3: Discuss the role of SSL in the TCP/IP stack and its implications for application developers who wish to enhance TCP with SSL. Consider the steps a developer must take, and the potential trade-offs involved in implementing SSL at different layers.

SSL is the abbreviation for Secure Sockets Layer. It helps in providing security over a TCP/IP stack by encrypting the data sent between the client and server. It also helps maintain confidentiality, integrity and authenticity. It is also known by TLS (Transport Layer Security).It operates between the transport and application layer .SSL encrypts data which is then sent to TCP for package transfer thus it enhances the features of TCP.
For an application developer to implement SSL he needs to take care of the following factors:-

1. **Layer Selection for SSL Implementation**

   - **Application layer:** SSL/TLS is most commonly implemented at the application layer (e.g., HTTPS for web applications). It uses libraries like OpenSSL. This leads to easier integration using well-established libraries, widespread support, and easier debugging.
     Trade-offs: Tightly couples security with the application protocol, potentially limiting flexibility.

   - **Transport Layer:** Implementing SSL/TLS directly at the transport layer is rare and complex and often needs customized implementations. This approach might be used for applications where developers want to increase security at a lower level or need to secure non-standard protocols. This may provide a more general security layer that can be reused across different applications or protocols.
     Trade-offs: More complex to implement, requires deeper understanding of TCP/IP stack, and harder to debug.

2. **Integration of SSL/TLS Libraries**
   Developers typically rely on libraries such as OpenSSL, GnuTLS, or the built-in TLS support in programming languages (like Java's javax.net.ssl package).This requires selecting the library that meets the security requirements, implementing techniques to handle certificates and finally configuring the SSL content(including cipher suites, protocols and session settings).
   Trade-offs: A stronger encryption obviously increases the CPU overhead. While selecting a library the developer has to choose between complexity and control i.e. higher-level libraries simplify integration but might hide critical security details but lower-level libraries offer more control but require more expertise.

3. **Handling SSL/TLS Handshake**
   During the SSL/TLS handshake, both parties exchange keys and negotiate encryption methods. This happens when the client initiates the handshake by connecting to the server over a secure port. Both parties then exchange public keys and agree on symmetric encryption keys. Session resumption mechanisms are implemented to reduce the overhead of repeated handshakes.
   Trade-offs: The application developer must select between security or latency as full handshakes offer the

highest security but introduce latency. Session resumption reduces latency but might be slightly less secure.

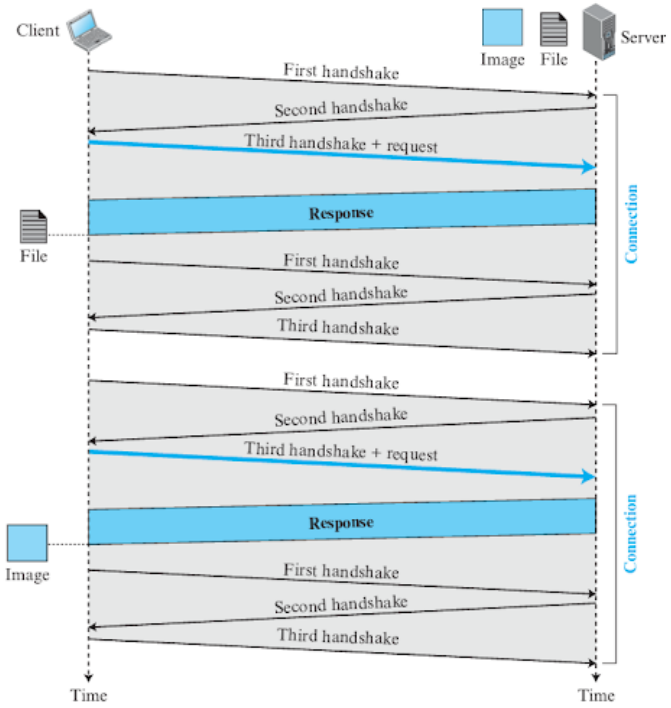
4. **Error Handling and Debugging**
   SSL/TLS are challenging to debug due to the encrypted nature of communication. Proper error handling is crucial to ensure robust security. The developer can implement detailed logging, especially for handshakes and certificate validation and also use tools like Wireshark and test suites to validate SSL behaviour.
   Trade-offs: Detailed logs can expose sensitive information if not properly secured. Keeping up with security updates and patches for SSL/TLS libraries adds to the development burden.
   Thus these are the factors that the application developers have to keep in mind while implementing the SSL into the TCP/IP stack.

Question 4: Examine the process by which non-persistent HTTP connections signal the end of a message to the TCP protocol. Discuss how this mechanism affects the performance and reliability of web communications, and compare it with persistent HTTP connections.

HTTP or HyperText Transfer Protocol is an application layer protocol which defines how web clients request web pages from web servers and how web servers provide the response.It basically tells us about the sequence and format of messages between the client and the server.

The HTTP uses the services of TCP or Transmission Control Protocol in the transport layer. Therefore, HTTP simply adds the service of pushing the message down the necessary socket and receiving a message from the socket whereas transport of the messages is governed by the TCP.

However, the application developer can decide up to some extent which features of transport layer protocol would be used.



Non persistent HTTP Connection: In a non persistent connection, a new TCP connection must be established between the client and the server for every request/response pair. The steps involved are

1. The client opens a TCP connection and sends a request.
2. The server sends the response and closes the connection.
3. The client reads the data until it encounters an end-of-file marker; it then closes the connection.

These steps are actually carried out for each individual object of the web page. For eg: If a web page has 10 images then it is actually made up of 11 objects(1 base HTML file + 10 images).Now, one request/ response pair requires typically 2 RTT(Round Trip Time) including the "three way-handshake" to establish a new connection and getting actual response. So using non-persistent HTTP for a webpage with N objects will take 2*N RTT to get us the complete webpage.

The HTTP server closes its side of connection after transmitting its entire response.However, to close the connection both sides must terminate it.

Now, when to end a connection can be specifically known by the HTTP client by either using the Content-Length header or the chunked transfer encoding. The Content Length header specifies the amount of data sent in bytes. Once the client gets that amount of data, it signals the end of connection from its side.

If chunked transfer encoding is used then the message is actually sent in chunks.Each chunk is preceded by its size in hexadecimal format followed by the data and then carriage return and new line. A zero-length chunk would mean an end of response and be a signal for the client to close its side of the connection.

Persistent HTTP Connection: In persistent connections, a TCP connection once established is used to send multiple request/response pairs .The server leaves the connection open for more requests after sending a response. The server can close the connection at the request of a client or if a time-out has been reached. Now, this takes 2RTT time for establishing connection and sending the first object and 1RTT for each subsequent object sent, so the total time required here is (N+1)RTT for a web page having N objects.

Advantages of Non-persistent connections:

- Non-persistent connections utilize network resources efficiently as wastage of resources is very less because connections are established only when objects must be transferred unlike persistent connections which may sit idle for longer times sometimes.
- They are more secure since connections close once the data has been sent.

Disadvantages of Non-persistent connections:

- Non-persistent connections require a lot of memory and processing overhead as the same handshake procedure must be carried out for each object.
- They take more time than persistent connections due to this overhead cost and hence have higher latency.
- They have relatively higher network congestion due to higher number of connections.

As of HTTP 1.1 and later persistent connections are made default.