Computer Systems 2

CEN 502

Project 2 Report

Group Members:

Mayank Gupta

ASU ID: 1209469518

Yogesh Kubal

ASU ID: 1209536364

Aman Karnik

ASU ID: 1209536091

INDEX

1. Problem Statement:

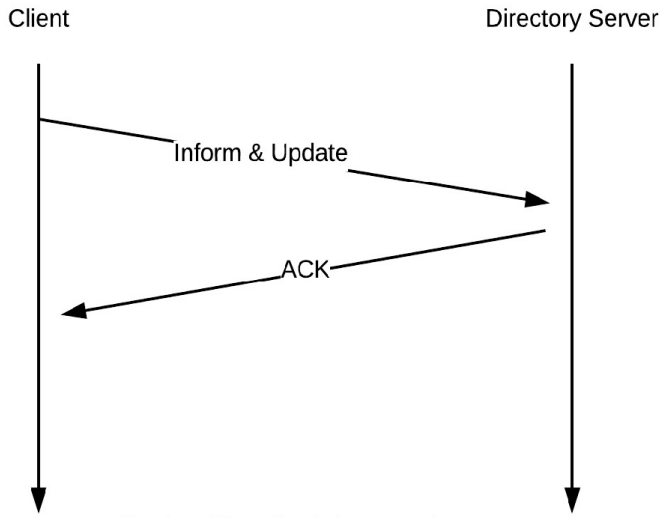
The objective of the project is to create a simple peer-to-peer (P2P) file sharing system. The system should consist of one centralized directory server and at least two peers. Each peer can combine the functionality of a P2P client and a transient P2P server. As a P2P client a peer can make requests to obtain files from directory server and obtain a list of files that the peers currently possess. The transient P2P server will serve files requested from a P2P client.

The two peers and the directory server are identified by a host name, a port number and each host will have a unique IP address.

2. Design:

The P2P file sharing system implemented consists of one centralized directory server and multiple number of P2P clients. The directory server acts as a database which keeps track of all the files that are held by the peers. When a client connects to a server, the client has three options at his disposal which are as follows:

i. Inform and update: When a new client joins into an existing network it can update the server by listing a set of files it wishes to share with the other peers in the network. Any client can connect to the server and update the database at any frame of time. The data packet flow for the Inform and Update request can be shown as:



Packet Flow for Inform and Update

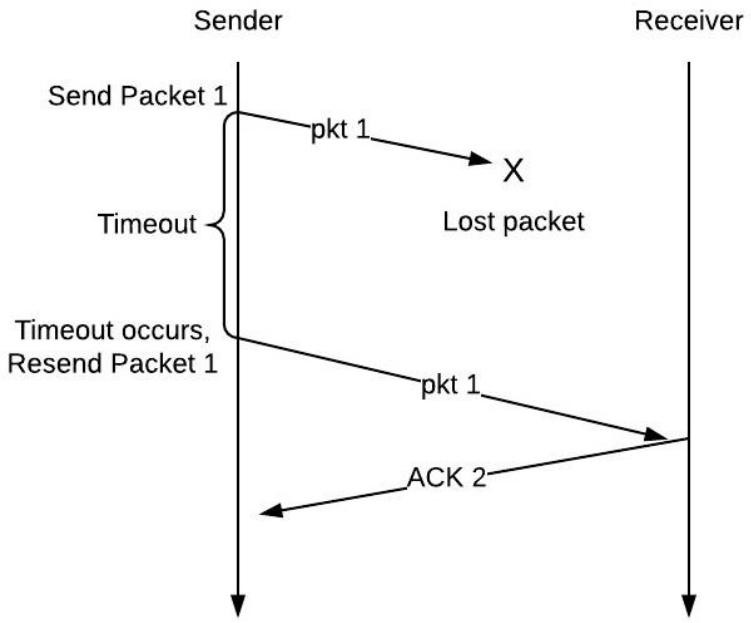
ii. Query for content: A client can send a directed request to the server for a particular file wherein the server will then provide the client with a list of all the clients who possess the requested file/object. The peers can then connect to other peers in the network and download the file/object.

iii. Exit: Upon inputting the exit choice the client terminates its connection with the server.

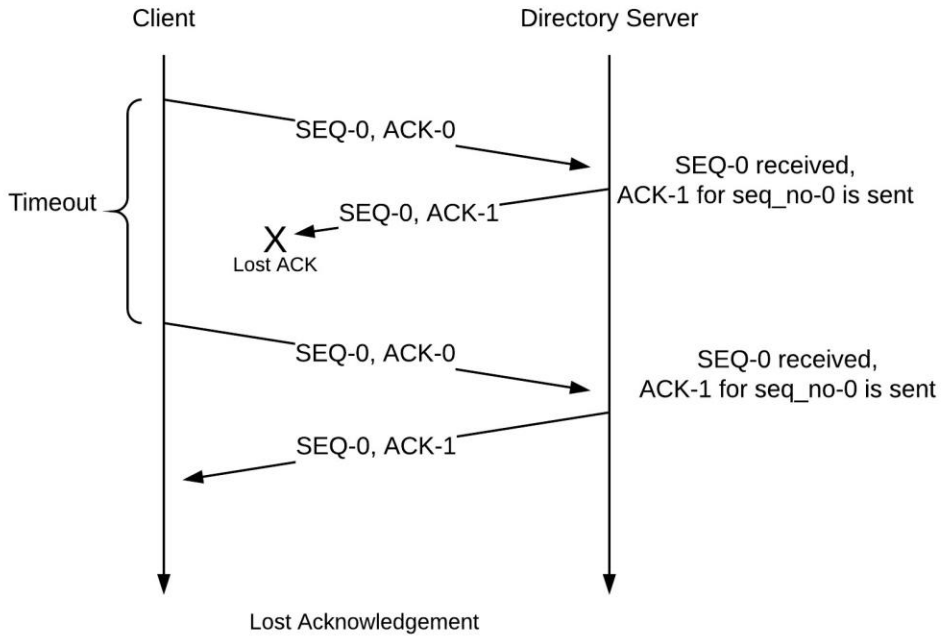
3. Features Implemented:

To improve the overall functionality of the P2P file sharing some features are implemented in the system:

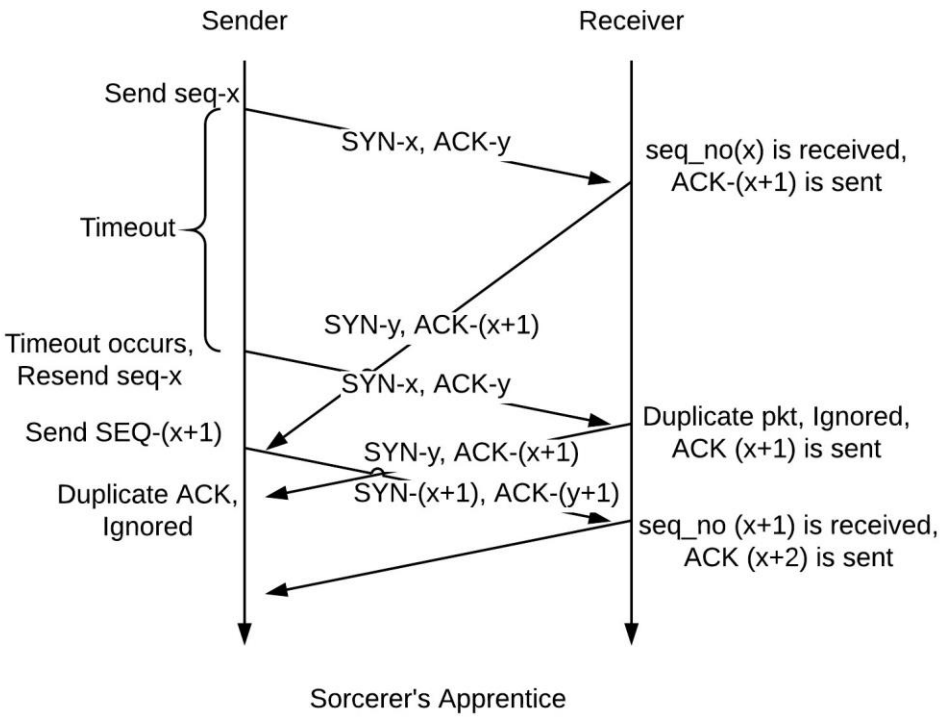
i. Timeouts: The timeout functionality is introduced such that when a packet transmitted on the sending side drops or gets misplaced during transit doesn't hinder the overall flow of the protocol. The sender will wait for a defined amount of time waiting for the receiver to send an ACK to the packet sent. Since the packet never reaches the destination no ACK returns. Thus after the wait time interval elapses the sender resends the same packet and the process continues.
The entire flow can be explained using the following diagram:



In a special case that the packet gets transmitted from the client and reaches the server but the resulting ACK gets lost. In this scenario the client waits for a receiving ACK. After waiting for a particular set time the client times out and resends the packet thinking the intended packet never reached its destination.



ii. Duplicate ACK's: The case for handling duplicate acknowledgements has been accounted for in the P2P file sharing system. Suppose a packet having sequence x is sent and the resulting ACK for sequence x has been delayed. Now the resulting ACK arrives but it arrives at the sender's side after the time out period. The sender should ideally resend the packet and the ACK will be returned by the receiver. Now since the sender already received an ACK it should discard this duplicate acknowledgement. Otherwise all further packets will be transmitted twice. This particular phenomenon in networking is called as Sorcerer's Apprentice.

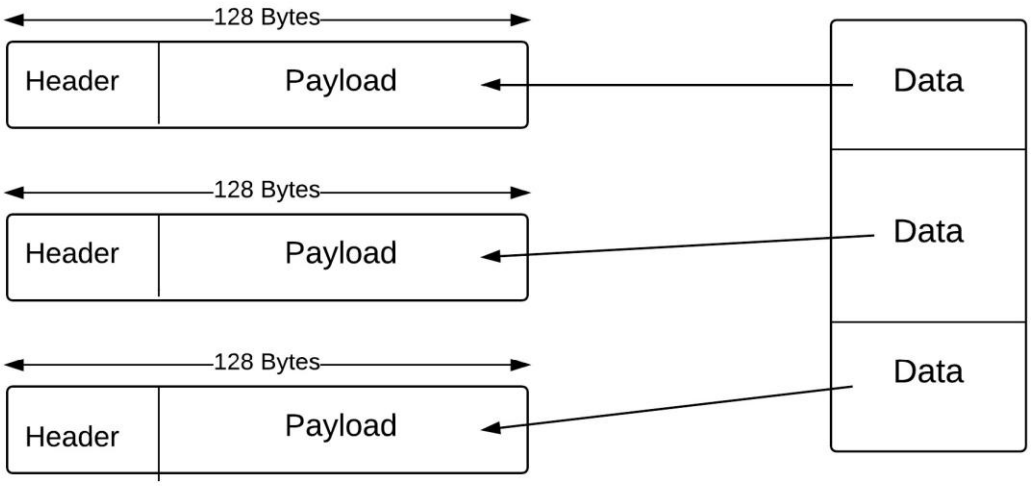


Sorcerer's Apprentice

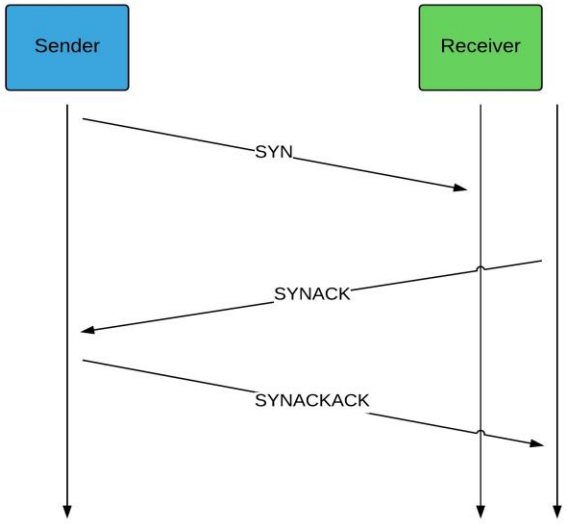
iii. Database using SQL: The central directory server acts as a database implemented in SQL, which holds the listing of files held by the clients in the network. The database contains columns for host name, files held by host, size of the file and IP address.

a. Every time the client inform and Updates the server, the new entry is logged into the database. We have written a SQL Insert query for adding this new entry into the database.
b. On Query for content, client will request a file. The server will run SQL Search query to go through the database and all the entries that match with the requested file will be returned by the server
c. On Exit, server will run the SQL Delete query, which will delete all the entries related to that exiting node.

iv. Packet splitting: Whenever the data file which is to be transmitted exceeds 128 bytes, the file is split into smaller packets such that the packet will contain maximum of 128 bytes including header and payload. An end flag has been implemented to signify the end of the packet transmission.



v. TCP handshake: Since UDP is a connectionless, unreliable protocol, the functionality of TCP is integrated in our P2P file sharing system. TCP is a connection oriented, reliable protocol which functions using three way handshake mode.



3 way TCP Handshaking over UDP

The client sends a SYN request to establish a connection to the server, the server responds using a SYNACK acknowledgement response and the client then sends a SYNACKACK which basically indicates that the connection is established. All further requests from the client to the server can contain the payload fields and data can be transmitted and requested.

4. Major Design Decisions:

I. Message formats:

Method 1)

Inform and Append

HEADER

“I&A ” +hostname+” “+Address+”\r\n”+END_FLAG+” “ + SEQ_NO. + “ “ + ACK_NO. + “\r\n\r\n”

+

BODY

Method 2)

Query

“QUERY ”+hostname+” “+Address+”\r\n”+filename+”\r\n”

Method 3)

EXIT

II.Pseudo-Code/Protocol:

P2P Client

1) Main function spins off a P2P Server Daemon

2) Ask user to choose one among the three method

3) User enters a filename and then next filename and then next till he adds all the files he/she wishes to add. The user may then type 'q' to finish adding.

4) The Client then builds packets whose structure is described above by fragmenting the string generated by concatenating 'hostname filename filesize Address' fields corresponding to each filename Appended) by the user to the database.

5) TCP Hand-Shake is done.

6) The packets are then passed on to the Directory Server in a sequential order. The sequence nos. for both Server and P2P Client are initialized randomly by each one themselves.

The pseudo-code for packet exchange is:

```
While(packets remain):
Success = 0
While(Success=0):
    Send_packet()
    Listen with timeout
    If ACK received:
        If !Duplicate:
            Success = 1
        IF Duplicate:
            Wait and listen for remaining time
```

7) Quering is done using the packet structure described above. While Quering if User enters 'LIST' as the filename. Directory server returns the whole database till now.

8) Post Quering User is also asked if he/she would now want to download any file that he/she wishes

9) The user must enter file name and the address of the Peer whose info it extracted using Query

10) If Requested a file by the name of copy_original_name is created in the Users present working directory

11) The user can choose option 3 if he wishes and then all his records would be deleted from the database

12) If not leaving User is again presented with the option of choosing :

a. Inform and Update

b. Query

c. Exit

Directory Server:

1) In the main loop Directory Server initializes a table by the name of “Listing” if not already initialized.

2) It then recursively listens for any incoming connections and everytime it receives a SYN packet that completes the TCP handshake using one of the free ports on the Computer running Directory Server

3) It then checks the subsequent packets for method Type and chooses to do one or the other based on that

4) Case: I&A

Sends an ACK everytime it receives a packet and also checks for Duplicate packets by comparing past ack_no and current seq_no.

5) Concatenates the string so that it now contains multiple entries of the form - ‘hostname filename filesize Address’

6) Passes the string to the database function that parses it and updates the database

7) Case: Query

Sends a query to the database and gets one or multiple entries depending on the request. If LIST then all entries . Otherwise it returns all the entries that match the filename.

The message fragmentation and sending part is exactly similar to the Peer’s

8) Case: EXIT

Deletes all the database entries related to the hostname and closes the connection

5. Conclusion:

Centralized P2P architectures have been in industry for a considerable time. One of the most established protocol utilizing this architecture is the Direct Connect protocol. Centralized systems are very efficient at reducing the resource location time and optimizing the overall network topology. Contrasting on these advantages, a centralized system focuses and prioritizes the server thereby making it an important entity in the entire network. Thus if the server fails, it creates a single point of failure. Moreover the server houses the information about the peers and the corresponding IP addresses, thereby if the server is victim of a malicious attack, valuable data is compromised.

A simple P2P file sharing system has been implemented and tested using a central directory server and two client peers. The entire communication has been monitored using the three message formats Inform and Update, Query for content and lastly Exit for quitting the communication link. The essence of P2P file sharing i.e. file transfer between the two peers using a central server has been successfully implemented.