CS3103 GROUP 3

Protocol Design

Brehmer Chan Xi Quan	A0146752B
Lim Wei Jie	A0139128A
Madasamy Ravi Nadar Mamtha	A0142073R
Marcus Ng Wen Jian	A0139257X

Overview

We aim to implement a P2P file transfer system with the architecture and design mentioned below.

Specifications

Programming Language

We coded in Java and used TCP for all connections due to scalability and ease of usability.

Role of Various Components in the System

Since the number of peers participating and size of uploaded data is small, we adopted a hybrid P2P network system, with a centralised server and few peers.

Role of Centralised Server (Tracker)

A tracker refers to the centralised server used to bootstrap a P2P system.

The functions of the tracker is to keep up-to-date information of the type of file chunks owned by each active peer, update information such as if the peers are still active or online and route to find peers with desired information.

The tracker maintains a hash table where each available filename is mapped to a list of peers with the chunks they have. Each entry in the tracker contains 4 items: chunk number, max chunk number of a file, ip address of the host where the chunk resides and port number in which the socket was opened. The tracker is multi-threaded to handle multiple writings in parallel and is always online.

Role of Peer

Each peer is identified by its unique pair of ip address and port number. The peer connects to other peers, through the TURN server inside the tracker, to download data. The peer also act as a P2P server to share chunks with others. Each peer has 2 persistent connections, one as a client and the other as a server. The client's persistent connection is to run various commands.

System Architecture and Design Protocols

Initialising

As shown in Figure 1, once Peer A enters the system, it establishes a persistent connection with the tracker for passive listening as a server. This is to receive any data transmitted from the TURN server. This initialisation would help to update the TURN server of the public IP address and port number of the peer, which is behind a symmetric NAT router.

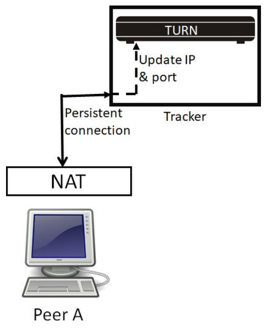


Figure 1

Downloading

Peer B would like to download abc.txt file. Firstly, Peer B would query the tracker using the client persistent connection, asking for the list of peers containing the chunks of abc.txt. Tracker would return with the ip address and port number of the various peers containing the abc.txt chunks. Peer B will randomly choose a peer from the list(Eg. Peer A) and open up a temporary socket, connecting to the TURN server, to request for connection with Peer A. The TURN server would then, ask Peer A to open a temporary socket through the persistent server socket with Peer A. Data will flow from Peer A to Peer B, through their respective temporary sockets, via the TURN server. After complete download of abc.txt by Peer B, Peer B informs tracker of its new available file. The temporary sockets of Peer A and B will then be closed respectively.

Informing

A peer can have new data or file in these scenarios:

1. The peer just joined the network and already has some data
2. The peer(eg. Peer B as explained above) downloaded from other P2P servers (eg. Peer A as mentioned above)

In such scenarios, the peer will inform the tracker, via the client persistent socket, to let the tracker know of its available chunks.

HeartBeat Signal

For the tracker to keep track if the peers are alive, each peer will send a heartbeat signal to the tracker at regular intervals via a separate thread. The tracker will compare the list of active peers with the list of all peers in the network and remove inactive peers.

Exiting

If the tracker does not receive a heartbeat signal or the peer send a command to exit the system, the tracker removes all the existing data related to that peer and closes all connections with that peer.

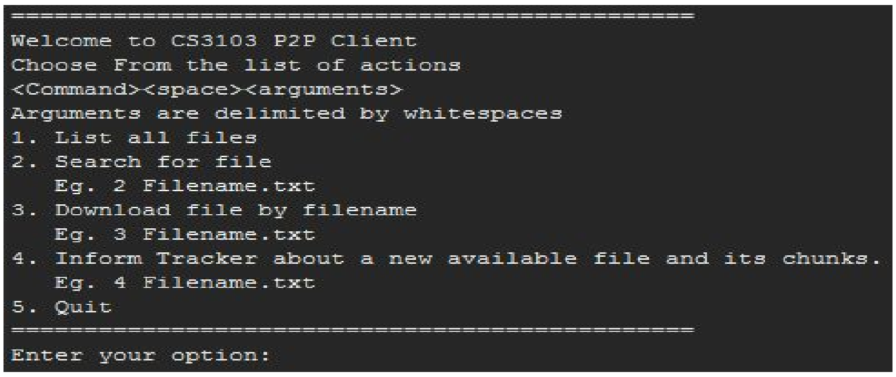
Message Format for Various Types Of Communication

Command	Peer to Tracker	Tracker to Peer	Peer to Peer
LIST	Request for list of files	Send list of files	-
SEARCH	Request to find file	Reply with boolean(ie. if file is available)	-
INFORM	Inform new available file	Acknowledged that it has received	-
DOWNLOAD	Request to download Request to download from peer	Reply with list of files and peers Relay data from destination peer to source peer	Reply successfully sent (ie. ACK)
QUIT	Request to leave	Close all sockets	-
HEARTBEAT	Send pulse at regular intervals	-	-

Instructions to run programs

Tracker
Obtain a runnable jar file (Tracker.jar) from source code (Eclipse -> File -> Export -> Runnable jar)
Upload jar file onto VM (Eg. Digital ocean)
Run jar file using `java -jar <Tracker.jar>`

Client
Obtain a runnable jar file (Peer.jar) from source code (Eclipse -> File -> Export -> Runnable jar)
Run jar file using `java -jar <Peer.jar>` on Windows Command Prompt Console.



```
==============================================
Welcome to CS3103 P2P Client
Choose From the list of actions
<Command><space><arguments>
Arguments are delimited by whitespaces
1. List all files
2. Search for file
   Eg. 2 Filename.txt
3. Download file by filename
   Eg. 3 Filename.txt
4. Inform Tracker about a new available file and its chunks.
   Eg. 4 Filename.txt
5. Quit
==============================================
Enter your option:
```

Follow on-screen instructions

Assumptions

- Assume ideal environment as stated on piazza.
- While a peer is informing the tracker of a file, other clients may be allowed to download the informed chunks from that peer, but may result in incomplete file at the end of the download. In that case, simply try downloading again.