Project Report

Group 10 Monica Bhargavi Kodali(UFID:16783481),Chandra Devarakonda(UFID:90922981),Rishabh Das(UFID:05369273)

I. Problem Statement

To implement a P2P file sharing software similar to BitTorrent.

II. Introduction

Peer-to-peer file sharing system for file downloading which is similar to BitTorrent is implemented in Java. This is used for transferring large files which allows users to distribute chunks of files in a decentralized manner over the Internet. TCP is used as the transport layer protocol for exchanging chunks.

III. Description

A. Handshake

First a connection is established between two peers and each of the peers of the connection sends a handshake message to the other peer before sending any other messages. The handshake messages are verified by checking whether the handshake message is correct and the peerID is the expected peer.
Bitfield message is sent after handshake whose payload indicates the file chunks/pieces a particular peer has.

B. Choke and unchoke

Each peer uploads its pieces only to preferred neighbors and an optimistically unchoked neighbor. Each peer determines preferred neighbors every p seconds. Those preferred neighbours are unchoked. Choked message indicates the peer cannot receive chunks from other peers. Unchoke indicates the peer will be able to receive and send chunks. Preferred neighbors are selected based on the downloading rates among the peers that are interested in the peer's data. If a peer has a complete file, it determines preferred neighbors randomly among those that are interested in its data rather than comparing downloading rates. An optimistically unchoked neighbor is determined by the neighbor every m seconds.

C. Interested and not interested

A peer sends an 'interested' message to the neighbor if neighbor has some interesting pieces regardless of the connection state of choked or unchoked. Each peer maintains bitfields for all neighbors and updates them whenever it receives 'have' messages from its neighbors. A peer sends a 'not interested' message to the neighbor if the neighbor does not have any interesting pieces and also if the piece is received by the peer completely.

D. Request and piece

When a connection is unchoked, a peer sends a request message for requesting a piece that it does not have and has not requested from other neighbors. It receives a 'piece' message from the the peer that contains the actual piece. After completely downloading the piece, request messages are sent until the peer is choked and the above process continues.

IV. Procedure to execute project

To run the project, prerequisite is the following:
-Login to the remote machine and compile ”javac PeerProcess.java”
- To run PeerProcess in the remote machines, we wrote a wrapper. So if we start the wrapper, all the processes will be started in the remote machines.
- To connect to the remote machines via our local machines, we need to first generate the keys.
Follow the below to generate the keys:
1. Run ”ssh-keygen” - it will prompt for file name and passphrase, please don't have a passphrase.
2. Run ”ssh-copy-id -i your _ key username remote _ machine” - Copy the key to one of your remote machines.
- Login to all the remote machines using this key once before you start the wrapper using ”ssh -i your _ key username remote _ machine”. This will avoid the fingerprint prompt message once you start the wrapper.
- Copy the Common.config file, PeerInfo.config file and the actual file to be transferred in the corresponding peer directories under the folder ”project/”.
- In the Ssh wrapper file, please change the username to yours, projPath to where you run the program and 'pubKey' to your generated rsakey.
-Compile java files using ”javac *.java” in ”Group_10/”, ”Group_10/project”,”Group_10/src”,”Group_10/docs”
- To start the wrapper compile the Ssh.java program - ”javac Ssh.java”, and run the wrapper - ”java Ssh”.