Project 2: Bittorrent Client

-----------------------

Name: RAHUL PASUNURI

email: rahupasu@indiana.edu

Name: MANISH KUMAR

email:manivutt@indiana.edu

------------------------

This project implements the functionalities of a Bit torrent client, in which all the participating Leechers can download files in parallel from multiple Seeders and a Seeder allows incoming Leecher connections as long as the no. of connections are below the maximum limit of support.

Functionalities:

1) The program takes the torrent file name as input from the command line and automatically extracts the file related info and connects to the Peer that has the pieces related to that particular file.

2) Integrity of the file downloaded is checked by comparing the downloaded piece’s SHA1 hash (computed) with the corresponding hash piece obtained from the parsed Torrent file.

3) The program has the ability to handle multiple download connections from the leechers at the same time.

4) A leecher can connect to any of the seeders that are provided from the command line to download the file by following the steps according to Bit torrent specification.

Rules for Compiling the code:

There is a 'makefile' written in the project's main folder. Open a comman prompt, 'cd' to the project's main folder, and then execute 'make' command. This command creates the object files and the executables. The executable is created in the bin folder.

Usage:

bt-client [OPTIONS] file.torrent

-h Print this help screen

-b ip Bind to this ip for incoming connections,

Ports are selected automatically

-s save\_file Save the torrent in directory

save\_dir (dflt: .)

-l log\_file Save logs to log\_file (dflt: bt-client.log)

-p ip:port Instead of contacting the tracker for a peer list,

use this peer instead, ip:port (ip or hostname)

(include multiple -p for more than 1 peer)

-I id Set the node identifier to id, a hex string

(dflt: computed based on ip and port)

-v verbose, print additional verbose info

-f If the option is provided then the program will

Launched as a Seeder else it instantiates a leecher

Examples:

(For the below commands to work, the command prompt must 'cd' to the project's main folder)

1) ./bin/executable.out -v –f –b localhost download.mp3.torrent

This binds the server (also a seeder) to the local IP address and uploads the download.mp3 file to the connected leechers. Port number is selected automatically. Re-run the same program to create multiple instances of seeders.

2) ./bin/executable.out -h

This prints the help screen.

3) ./bin/executable.out -p 127.0.0.1:6668 -v download.mp3.torrent

This initializes the leecher that connects to a seeder listening on 127.0.0.1 and port no: 6668.

4) ./bin/executable.out -p 127.0.0.1:6668 –p 127.0.0.1:6669 -v download.mp3.torrent

This initializes a leecher having a seeder list(IP:Port combinations after -p) and establishes connections to them for downloading the torrent.

5) ./bin/executable.out -p 127.0.0.1:6668 –p 127.0.0.1:6669 –v –s <PATH> download.mp3.torrent

Using the -s option we address the program to save the downloaded file in the path provided.

Warnings:

1) If '-f' is given in the options list, we ignore all the leecher options, and run only the seeder. This is the distinguishing parameter we used to differentiate between seeders and leechers.

2) If '-b' is specified, then the server binds to the ip given using this option else it binds to localhost by default.

3) The program terminates if the specified torrent file does not exist.

4) If '-s' option is not provided, then the downloaded file will be saved in the current directory with the name “alpha”.ext

Description of Project Folders:

1) All the header files are placed in the 'include/' folder.

2) All the .cpp files are placed in the 'source/' folder.

3) The generated object files will be placed in the 'object/' folder.

4) The generated executable files will be placed in the 'bin/' folder.

5) The files received by the server are saved in the working directory if '-s' is not given.

Limitations of the Project:

1) Works only for IPv4 addresses.

2) The project developed doesn’t have the functionality to coonect to tracker. Hence the peer list must be manually given to the program using –p option when instantiating a leecher.