CSN-361

Computer Networks Lab

Group-6

Project Report

under

Prof. Sandeep Kumar Garg

Date: 10 November, 2024

Contributions

Name: Meet Sindhav
Roll No: 22114053
Email: meet_js@cs.iitr.ac.in
Mobile No: 7984221079
Contribution: Worked on implementation of client part of the project. Worked on establishing communications between peers and eventual downloading. Also worked on managing file pieces.

Name: Mohammed Haaziq
Roll No: 22114055
Email: mohammed_hj@cs.iitr.ac.in
Mobile No: 8126061554
Contribution: Worked on handling of connect and announce requests by the peer.Also on writing functions related to storing data about leechers and seeders in database using the sqlite3 database.

Name: Aditya Mundada
Roll No: 22114058
Email: mundada_am@cs.iitr.ac.in
Mobile No: 7720079142
Contribution: Worked on implementing UDP tracker: handling connection requests and announce requests, establishing client-tracker communication. Also, involved in overall code clarification.

Name: Nayan Kakade
Roll No: 22114060
Email: nayan_rk@cs.iitr.ac.in
Mobile No: 7420936972
Contribution: Worked on parsing of the torrent file using Bencoding Library. Also worked on connection of peer and tracker which included sending connect request and announce request to tracker and retrieving list of peers as response.

Name: Sarvesh Rupesh Baheti
Roll No: 22114087
Email: sarvesh_rb@cs.iitr.ac.in
Mobile No: 9767547468
Contribution: Worked on socket utilities that are required between tracker and peer communication. Also, on setting up the UDP tracker.

Name: Roopam Taneja
Roll No: 22125030
Email: roopam_t@cs.iitr.ac.in
Mobile No: 9810788629
Contribution: Implemented cutom bencoding library. Also worked on implementing client part. Managed communication with multiple peers and succesful downloading of files.

OurTorrent

Overview

The **aim** of our project is to build **OurTorrent**: a Bittorrent inspired P2P file sharing system which allows downloading files from a distributed set of peers rather than a central server.This helps users of the system download files faster due to multiple uploaders present at the same time rather than a single central server serving numerous requests.

We took inspiration from the BitTorrent protocol, also referred to simply as torrent, a communication protocol for peer-to-peer file sharing, which enables users to distribute data and electronic files over the Internet in a decentralized manner. It involves oversight of the peers, pieces (file is divided into parts called pieces), connections among peers, seeding, downloading etc. We tried implementing a client as well as a minimal UDP tracker to gain a complete understanding of the protocol.

We built the project in two phases:

1. **Phase 1**: It involved implementing a client for the Bittorrent protocol. This client of ours aims to use the torrent file corresponding to any file we wish to download from the internet.Our client then can access any general tracker - the one whose URL is mentioned in the Torrent file, present on the internet to connect to that group of peers and be a part of that torrent. We can thus download any file using our own client.

2. **Phase 2**: In this phase we tried implementing our own bittorrent tracker to understand the nuances behind it.Our tracker can store the necessary attributes corresponding to each peer connected with us and then send information like peer id,seeders,leechers and other information related to who owns the necessary file to any of the clients that requests it.

Motivation

We chose to build this system because we were academically interested in understanding the very basics of how communication between multiple devices in a network is implemented. The best way to understand this was by implementing some sort of peer to peer system where we can simulate this communication albeit at a small scale.

Bittorrent helps us explore and study the underlying network protocols,programming languages and its popularity helps us make any significant improvements create farther reaching impacts. Keeping this in mind, we tried out a few improvements in the Original Bittorrent Protocol as mentioned in Bittorrent Extension Protocol(BEPs) such as implementing a lightweight UDP tracker instead of a conventional TCP one to reduce overheads.

Thus, our system can be optimized further and can be used as a black box to build up applications in the future.

Also, it can be used by a small group of our own friends to share large files among ourselves like games, movies and music.

Key Features

The key features of our system:

1. **Implementation of a UDP tracker**: Our project implements a UDP tracker which is lightweight without significant overheads and stores as well as dynamically updates each piece of necessary information related to a peer currently forming a part of our sharing community. It provides any client with all the necessary information about peers that contain the file which the client requires.

2. **Asynchronous socket handling**: In order to handle multiple clients to tracker as well as multiple client-client connections , we implemented asynchronous functions for efficiently managing these connections.

3. **Bencoding Serialization library**: We have also created our very own serialization library which serializes the data and converts it to the necessary format that can be sent over to other peers as well as to deserialize responses.

4. **Error handling**: We have taken care of various errors that may arise during this communication like response timeouts and implemented mechanisms to take care of such issues.

5. **Security**: We have managed a rudimentary layer of security by ensuring that only the requesting peer gets the necessary information even within the UDP structure by implementing Connection IDs for each peer.

Implementaion

1. **Implementation of a UDP-based tracker**:

    - The tracker initializes the database made using SQLite3 in its main function.
    - Then a client has to parse its torrent file to extract different fields like tracker URL, info hash, and other metadata.
    - The tracker uses UDP communication protocol to interact with clients.
    - The client has to send a connection request to the tracker, functioning as a handshake upon which the tracker would send a connection response in return.
    - After this, the tracker inserts client information into the tracker database based on provided parameters, categorizing the client as either a seeder or leecher.
    - Upon receiving a connection response from the tracker, the client needs to send an announce request that includes parameters like transaction ID, peer ID, downloaded bytes, uploaded bytes, and more.
    - The tracker then retrieves peers from the tracker database based on client requests.
    - It responds to client announce requests with this list of peers, each entry including the IP address, port, and peer ID of the respective peer.

2. **Bencoding library**:

- A small, single-header serialization library for bencoded data.
- Bencoding is used by Bittorrent protocol for encoding metadata of .torrent files
- The library supports serializing and deserializing bencoded data with a simple API written specifically to support the project.

3. **Bittorrent client**:
   - Communicates with a tracker on the Internet to retrieve list of peers and tries to establish connection with them.
   - Initializes handshake with each peer and waits for them to unchoke the connection.
   - Once unchoked, client is able to download its pieces.
   - This allows to download a file from any corner of the Internet just from its torrent file.

Tools and Technologies Used

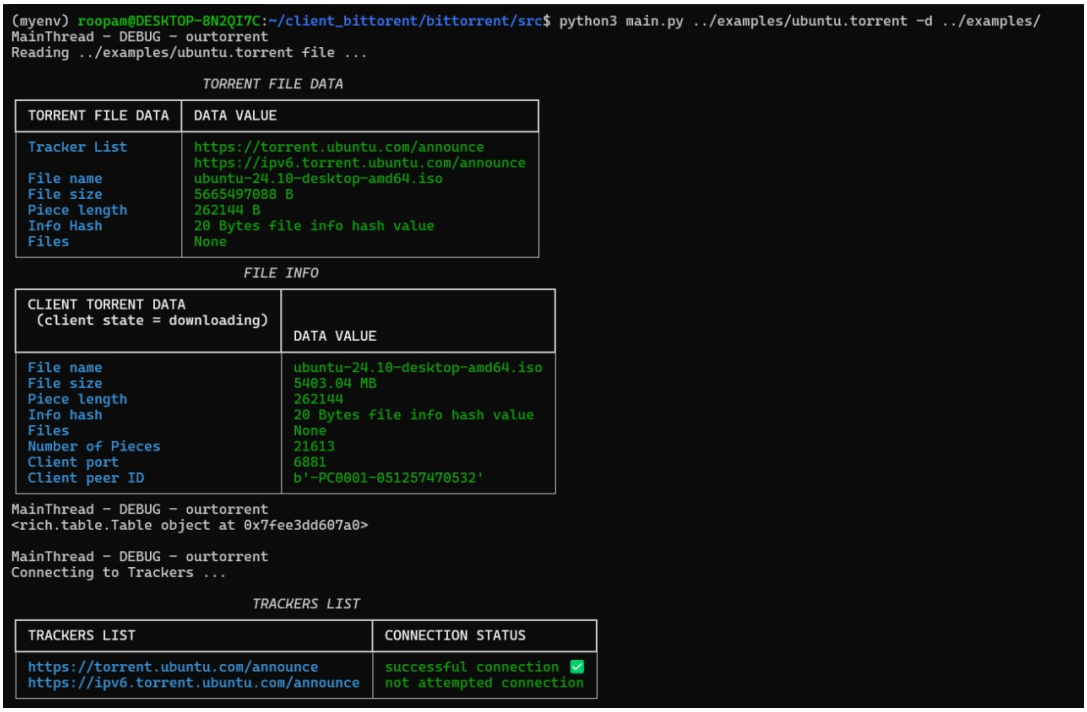
- Sqlite3 for database
- POSIX socket libraries
- OpenSSL for SHA1 hash
- Rich package for table formatting
- urrlib3 library for HTTP connections

Applications and Innovations

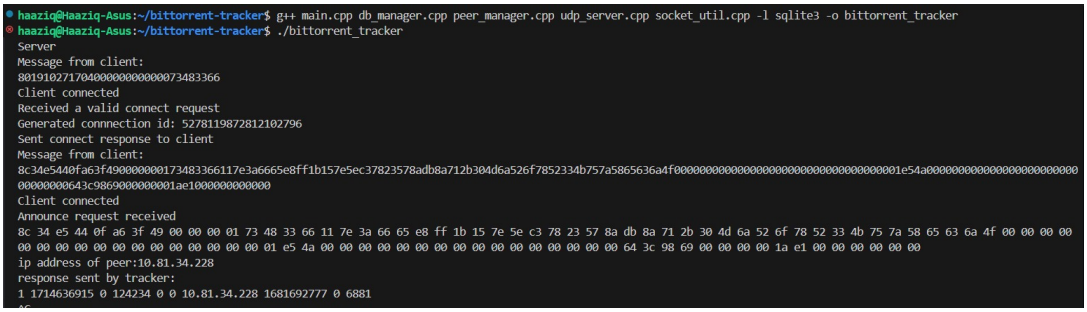
1. **File Sharing**: BitTorrent is widely used for distributing large files such as videos, software, games, and operating system images. It's a popular method for distributing Linux distributions, like Ubuntu, because it reduces server load.
2. **Decentralized Backup Systems**: Some organizations and individuals use BitTorrent for decentralized backup systems, where files are shared across many peers for redundancy and fault tolerance. This approach ensures that a file remains accessible even if some nodes fail.
3. **Software Distribution**: Game companies like Blizzard (creators of World of Warcraft) use BitTorrent technology to distribute game updates and patches to players, reducing server strain. BitTorrent is sometimes used for distributing large software updates or packages, especially for open-source software. Blockchain systems like Bitcoin and Ethereum use a similar P2P architecture to distribute and synchronize node data.
4. **Video Streaming**: Some platforms (e.g., Popcorn Time) have adapted BitTorrent for streaming video content. Instead of downloading an entire file before watching, users stream content as they download it in chunks from other peers.

Screenshots of the System

1. **Client Download Interface**:



2. **Tracker's Interface**:

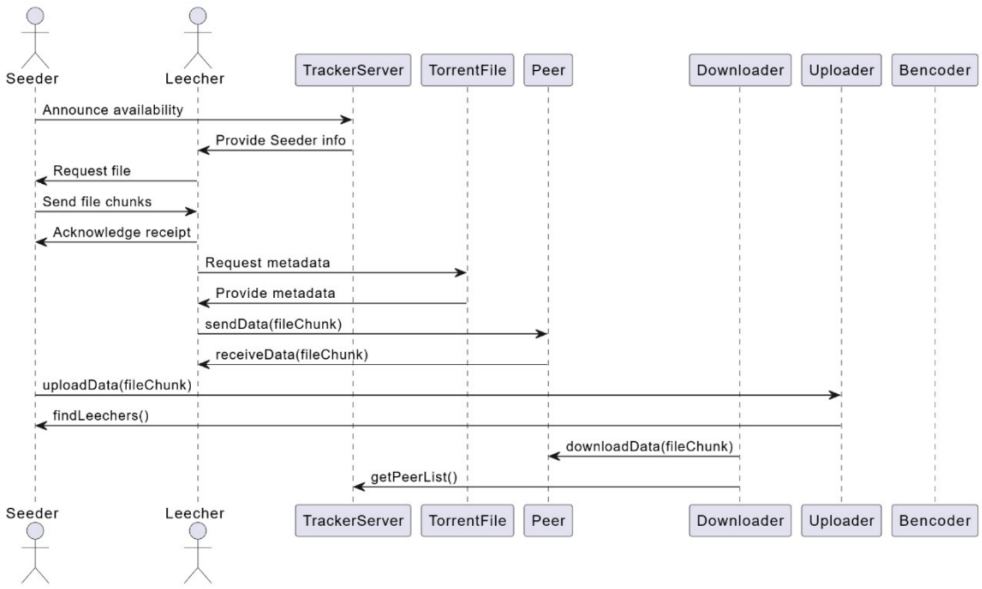


3. **Tester's Interface**:

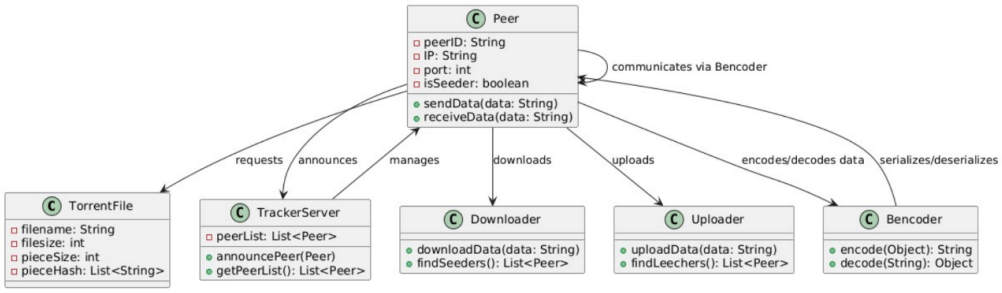
```
Transaction ID: 1714636915
Sent connect request to tracker.
Got packet from tracker
Packet is 16 bytes long
Received valid connection ID: 5278119872812102796
Sent announce request to tracker.
Received packets from tracker
Packet is 158 bytes long


Peers list is as follows:

IP: 10.81.71.27, Port: 6881
IP: 192.168.12.4, Port: 48967
IP: 192.168.31.8, Port: 6969
IP: 10.81.72.3, Port: 32428
nayan@nayan-Inspiron-3501:~/Desktop/BitTorrent$
```

Diagrams

1. **Sequence Diagram**:



2. **Class Diagram**:



Future Scope

1. Implementing better security standards using public and private key encryption methods and identifying spam users.
2. Implementing other Bittorrent extensions as mentioned in BEPs like Distributed Hash Table protocols like chord.
3. Extending functionality to IPv6 to provide better functionality.
4. Extending scope to operating systems like Windows which are not UNIX based.
5. Implementing resource management mechanisms for peers.
6. Using even more sophisticated database management for increasing efficiency.

Conclusion

In conclusion, our project has successfully demonstrated the potential of a BitTorrent-inspired P2P file-sharing system, highlighting its effectiveness in decentralized file distribution. By leveraging peer-to-peer connections, our system enables users to download files from a distributed network of peers rather than a central server, enhancing both speed and reliability. Key features like the use of Bencoding for metadata sharing, tracker implementation, UDP for tracker communication, and TCP for peer-to-peer connections were instrumental in achieving robust functionality.

The system's strengths include efficient resource utilization and enhanced download speeds due to managing simultaneous connections. Our performance benchmarks reveal promising results in terms of download speed, memory usage, and connection management.

Despite the progress, we recognize areas for further improvement. Optimizing for IPv6 support, addressing potential security concerns, and reducing latency in high-traffic environments are vital for broader adoption and scalability. With these insights, we are motivated to enhance our P2P framework, allowing for versatile applications in real-world use cases and advancing our understanding of network protocols and distributed systems.

Source code

Github link of source code

References

1. BitTorrent Specification - Theory Wiki
2. BEP 3 - BitTorrent Protocol Specification
3. BitTorrent Overview - Brown University