

Assignment 1

SDE

Sireejaa Uppal

M23CSE023

17th September, 2023

1. System Design

1. Clearly define the architecture of your P2P system.

A Peer-to-Peer (P2P) network is a decentralized architecture where individual computers or devices, referred to as "peers," directly connect and share resources like files, data. Here, the need for a central server is eliminated. In a P2P network, each peer can act both as a client and a server, allowing for bidirectional resource sharing. In this designed Peer to Peer network, There are two types of nodes in the network. One is the smart node and rest all nodes connect as dumb nodes. The smart node keeps track of all the connected peers and the list of files they have.


Whenever a new peer enters, it contacts the smart node and is provided with the list of already connected peers in the network.

If a peer needs a file in the network, it enters the file name, and if any of the peer in the network has the file, it is shared to the peer which had requested for it. In case no peer contains the file which was requested then an appropriate message is displayed to the node saying that the requested file is not found. Note: make sure to enter the file extension while searching for the file name. In case the file is present at multiple nodes in the network, then it is broken into chunks and sent by various nodes to the node which had requested for it. The chunk size is 4096.

2. Describe how peers will connect, communicate, and share files.

When the system is started, there is only a smart node in the network. Whenever a new node wants to join the network, it initiates a connection with the smart peer. The smart peer logs its details and acknowledges the incoming of a new peer to rest all the peers in the network with the help of a broadcast message.

Once the connection is established, the node can request for a file. If it is present then it is shared in chunks over the network otherwise a message saying that no such file is found is displayed. Dumb peers share files by making them available in their local directories. When a peer wants to download a file, it can request the file details (e.g., size, name) from other peers. File transfer occurs in chunks, and peers can download chunks from multiple sources in parallel. The received file chunks are combined to reconstruct the complete file locally. Communication Protocol for the peers communicate using a



predefined protocol. Messages are serialized (e.g., using JSON or pickle) for transmission. Message types include requests for file details, file chunks, peer lists, and network updates. Peers can also communicate to update their status, share available files, or disconnect from the network.

3. Handling Challenges:

Peer Discovery : Implementing a central server, also known as a smart peer, can facilitate initial peer discovery. New peers can connect to the smart peer to announce their presence and request a list of connected peers. The smart peer maintains an updated directory of peers and their addresses, making it easier for new peers to join the network. Peers can periodically request updated peer lists from the smart peer or known peers to stay informed about changes in the network's topology.

Security: Security mechanisms, such as encryption (e.g., TLS/SSL), can be implemented for peer-to-peer communication to protect data in transit. Authentication and authorization methods can be used to verify the identity of peers before allowing them to connect and share files.

Peer Reputation: A reputation system where peers can rate and review each other based on their behavior. This can help identify trustworthy peers and discourage malicious ones.

2. Implementation(README)

1. Implement a working P2P system.

The code is attached. There are two python files.

Step1 : Run the smart.py file in the terminal. Once its up and running move to step2 2.

Step 2: Run the dumb.py file as many times as many number of nodes you want to generate.

The following screenshots show the creation of smart node and initialization of network:

```
308 print("Please choose an option to perform a f...")
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\ACER\Desktop\p2p_siri - Copy> python .\smart.py
***** Welcome to the Peer to Peer Network *****
No current connectionns availaible. Setup the network. Only the smart node exists as of now
```

The following screenshot shows creation of 2 peers a and b, with ports 123 and 125.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\ACER\Desktop\p2p_siri - Copy> python .\dumb.py
***** Welcome to the Network *****
In order to become a member of your Network Please enter Your Details
Please input your port number: 123
Please input your name: a
Hi Peer, your existing files are : ['siri.txt']
Please choose an option to perform a function according to the menu list given below :
***** Currently Connected Peers in the Network are [('127.0.0.1', 123)] *****
Enter 0 to exit the network
Enter 1 to become a part of this Peer to Peer Network
Enter 2 to find the already present nodes in the network
Enter 3 to search for a file and get it from the peers in the network
Enter 4 to see the list of files that I have which can be shared to other users in the network
Enter 5 to exit the program
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
***** Welcome to the Network *****
In order to become a member of your Network Please enter Your Details
Please input your port number: 125
Please input your name: b
Hi Peer, your existing files are : ['example.txt']
***** Currently Connected Peers in the Network are [('127.0.0.1', 123), ('127.0.0.1', 125)] *****
*
Please choose an option to perform a function according to the menu list given below :
Enter 0 to exit the network
Enter 1 to become a part of this Peer to Peer Network
Enter 2 to find the already present nodes in the network
Enter 3 to search for a file and get it from the peers in the network
Enter 4 to see the list of files that I have which can be shared to other users in the network
Enter 5 to exit the program
```

The node a contains a file called siri.txt and node be contains a file called example.txt as shown in the screenshot

```

***** Currently Connected Peers in the Network are [('127.0.0.1', 123)] *****
Enter 0 to exit the network
Enter 1 to become a part of this Peer to Peer Network
Enter 2 to find the already present nodes in the network
Enter 3 to search for a file and get it from the peers in the network
Enter 4 to see the list of files that I have which can be shared to other users in the network
Enter 5 to exit the program

>***** Currently Connected Peers in the Network are [('127.0.0.1', 123), ('127.0.0.1', 125)] *****
**
4
The files present at your end are: ['siri.txt']
>

```

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Please input your port number: 125
Please input your name: b
Hi Peer, your existing files are : ['example.txt']
***** Currently Connected Peers in the Network are [('127.0.0.1', 123), ('127.0.0.1', 125)] *****
*
Please choose an option to perform a function according to the menu list given below :
Enter 0 to exit the network
Enter 1 to become a part of this Peer to Peer Network
Enter 2 to find the already present nodes in the network
Enter 3 to search for a file and get it from the peers in the network
Enter 4 to see the list of files that I have which can be shared to other users in the network
Enter 5 to exit the program

>4
The files present at your end are: ['example.txt']
>

```

2. Peers should be able to join the network, share files to other peers.

Suppose B wants file siri.txt. It will select option 3 and enter the file name. The peer A will share the file as shown in the screenshot below:

```
PROBLEMS  CONSOLE  DEBUG CONSOLE  TERMINAL  FILES
Enter 0 to exit the network
Enter 1 to become a part of this Peer to Peer Network
Enter 2 to find the already present nodes in the network
Enter 3 to search for a file and get it from the peers in the network
Enter 4 to see the list of files that I have which can be shared to other users in the network
Enter 5 to exit the program

>4
The files present at your end are: ['example.txt']
>3
Enter file name : siri.txt
(FYI) I am fetching the chunk 0 of 1 from the peer('127.0.0.1', 123)
received siri.txt
>4
The files present at your end are: ['example.txt', 'siri.txt']
>|
```


3. Ensure appropriate error handling and user-friendly interactions.

The command line interface provides extremely user friendly option selectable menu to the user to choose from a variety of operations that it can perform in the network.

3.Efficiency and Scalability

- 1.Consideration of system performance as the number of peers increases.

In terms of Scalability, as the number of peers increase, both "smart.py" and "dumb.py" can benefit from scalability improvements. As the number of dumb peers increases, the smart peer's ability to manage and provide an updated list of peers may become a performance bottleneck. The smart peer's data structures can be changed to a hash table and lookup algorithms can be implemented to handle a larger peer population efficiently. As more peers join the network, network traffic can increase significantly, especially during file transfers and peer discovery. Implement efficient communication protocols to minimize unnecessary data transfers and reduce overall bandwidth usage. Consider strategies like data compression or differential updates to optimize data transmission. Search and Discovery: In "dumb.py," the peer discovery process relies on a central smart peer. To handle increased peer counts, optimization of the search and discovery mechanisms to maintain low lookup times is required. Efficient data



structures like DHTs or Bloom filters can help enhance the discovery process. This will also increase the latency in the system. But additional peers will also accompany additional data and files to better serve the peers which will increase the throughput and benefit for the users and serve as an incentive for the the peers which are a part of our paper to peer network.

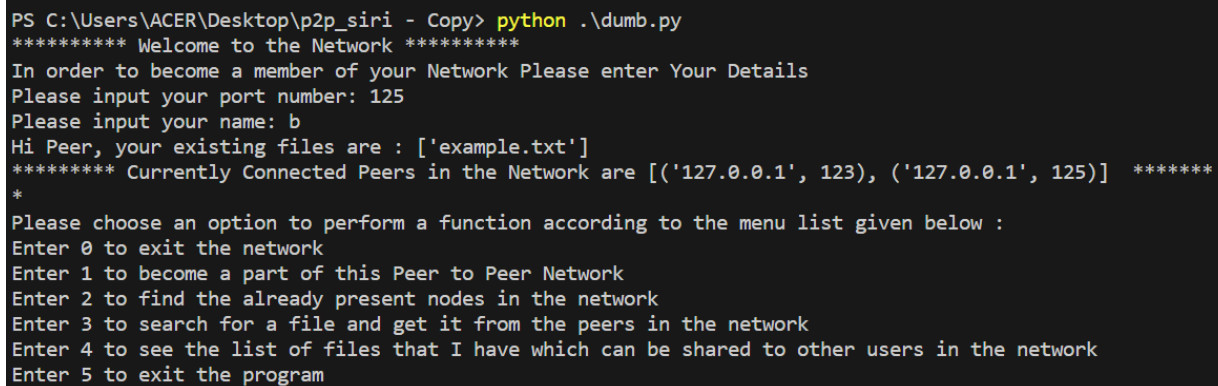
2. Discussion of strategies to ensure efficient file discovery and download.

The application of Distributed Hash Tables (DHTs) and Implementing a DHT-based indexing system, like BitTorrent will lead to more efficiency. DHTs enable decentralized and speedy file discovery by mapping files to unique keys. Users can query the DHT with the file's key to locate the nearest peers possessing that file. Caching is another way. Establish caching mechanisms for frequently accessed files. These caches can exist at various levels, from individual peers to network nodes. Frequently requested files become readily accessible, thus reducing download times. Query Optimization can also reduce the search time. Streamline query routing and cache query responses to mitigate redundant lookups. Load balancing can also expedite the process of searching and downloading files in case the peer to peer network is scaled to a huge network. Along with this prediction machine learning algorithms can be utilized to predict the files which a set of users are usually downloading based on repetitive patterns and consumer consumption study via ML algos.

4. User Interface:

1.Create a user interface that allows users to interact with the P2P system.

The command line interface provides a wide option selection to the user peer as shown in the screenshot below:



```
PS C:\Users\ACER\Desktop\p2p_siri - Copy> python .\dumb.py
***** Welcome to the Network *****
In order to become a member of your Network Please enter Your Details
Please input your port number: 125
Please input your name: b
Hi Peer, your existing files are : ['example.txt']
***** Currently Connected Peers in the Network are [('127.0.0.1', 123), ('127.0.0.1', 125)] *****
*
Please choose an option to perform a function according to the menu list given below :
Enter 0 to exit the network
Enter 1 to become a part of this Peer to Peer Network
Enter 2 to find the already present nodes in the network
Enter 3 to search for a file and get it from the peers in the network
Enter 4 to see the list of files that I have which can be shared to other users in the network
Enter 5 to exit the program
```

2.The interface should facilitate file sharing and downloading.

The interface allows the peer to select option 3 and enter the file name. If it is present with any other peer in the network it is transferred in chunks to this peer otherwise a file not found message is displayed.. The screenshot below shows the same:


```
PROBLEMS  CONSOLE  DEBUG CONSOLE  TERMINAL  FILES

Enter 0 to exit the network
Enter 1 to become a part of this Peer to Peer Network
Enter 2 to find the already present nodes in the network
Enter 3 to search for a file and get it from the peers in the network
Enter 4 to see the list of files that I have which can be shared to other users in the network
Enter 5 to exit the program

>4
The files present at your end are: ['example.txt']
>3
Enter file name : siri.txt
(FYI) I am fetching the chunk 0 of1 from the peer('127.0.0.1', 123)
received siri.txt
>4
The files present at your end are: ['example.txt', 'siri.txt']
>

Ln 350, Col 1 (12900 selected)  Spaces: 4  UTF-8  CRLF  (3) Python  3.1
```



5. Follow up Question:

1. Discuss the advantages and challenges of a decentralized P2P architecture compared to a hybrid architecture that involves some centralization.

A Decentralized Peer to peer architecture possesses no hierarchy and all the nodes are treated as equals with same rights. There is no single point of failure, thereby making the system highly resilient and fault tolerant. There is more privacy as there is no central authority to monitor and all the nodes are functioning independently. Also it is more difficult to attack a pure peer to peer network as there is no single point to attack which can bring the entire network down. The challenges include censorship and maintenance of authentication and reliability within the network.

On the other hand a hybrid architecture possesses improved resource discovery as compared to a pure peer to peer architecture. The data availability is enhanced in this case as well as frequently used files can be cached. This improves the performance and overall efficiency and throughput of the entire network. This architecture can also keep a control and check on the content, thereby maintaining law and order in the network. But due to hybrid centralization, there are scalability issues and breach of privacy may occur if the security is compromised.

