1.How to prevent other peers from filling up the disk?

There is a list of struct peerInfo, which maintains the total in_data, which is the data from a particular peer to the client and total out_data which is the data from client to a particular peer. This data is updated whenever there is a file GET or PUT.

There is a macro defined in the Constants.h for the MAX_DOWNLOAD_LIMIT which is always checked whenever a file is PUT by a peer.

If the PUT will make the total in_data > MAX_DOWNLOAD_LIMIT , then the PUT is rejected by client with a suitable reply.

2.How to prevent access from files in other directories in a GET?

Whenever a GET request is processed with a filename which contains "/", it is rejected. A "/" implies a access to some other directory and hence it is rejected. Only filenames without and path is accepted to be a valid filename.

It may be argued that <fullpath>/filename might be the directory under which the program is running but such path should not be known by the other peers. Hence by simply taking only filename(without paths) can solve the problem.

3. Working of SYNC

Whenever a sync is issued by client, it is passed to the server and the server broadcasts it to all the clients.

The clients the call the executeSync().

executeSync()

{

- Exchanges the filesizes of <Server>.txt files //as stated in statement for simplicity - the program just takes care of the <server>.txt files only
- Creates the file handles for all the peers connected to it.

While(1)

{

- Loops over all the peer and send() one buffer read of the own file to all peers
- After the send loop is over it loops again for all peers to to recv() from all the peers.
- Checks if all the data is sent and received and breaks

}

}

All the Clients simultaneously execute the executeSync() function and recv() and send() data in parallel.