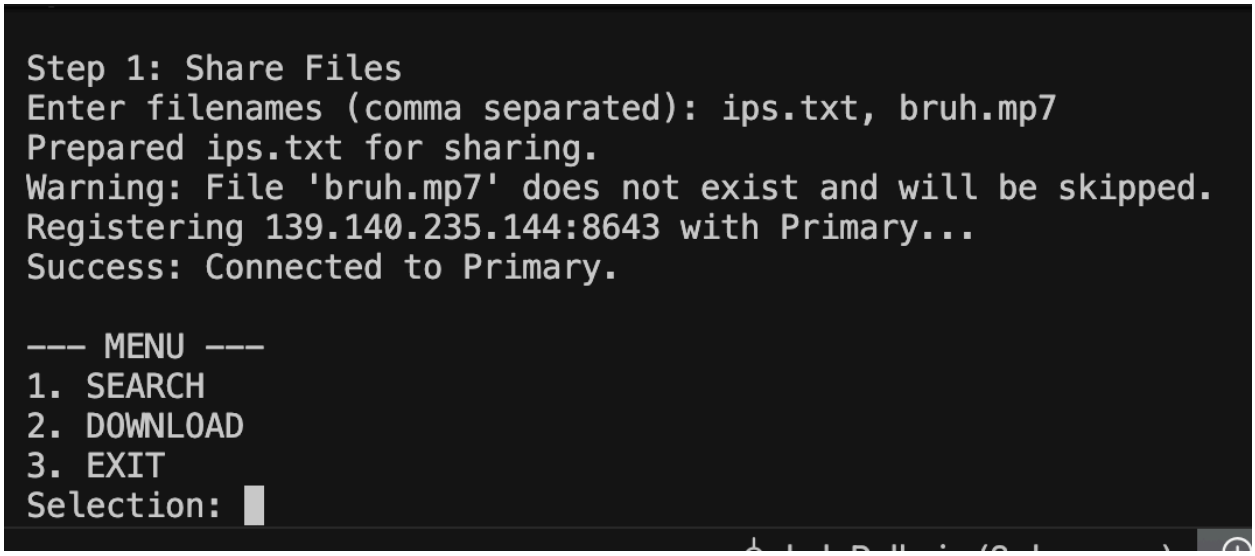
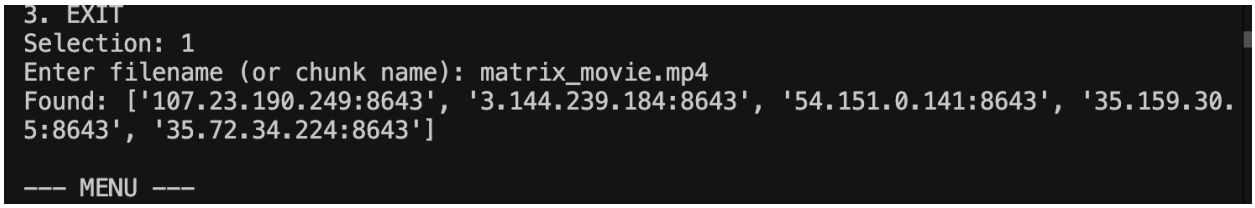
1) To first start run sync.sh user specified (change your username in the .sh file) to ensure servers are up to date with the latest changes
2) Then ssh a new terminal to **54.226.158.73,** which will serve as the backup server.
3) Then ssh a new terminal to ssh **54.205.35.150,** which will serve as the main server, notice that on init it will not have any files copied over from backup
4) In the backup server terminal run **java -cp ".:lib/*" MainServer 8642** to start the backup
5) In the main server terminal run **java -cp ".:lib/*" MainServer 8641 54.226.158.73 8642** to start the main server with the specified backup
6) Now in the original terminal, run the swarm.sh file, this will launch a swarm of 5 peers that each have their own data# file and a communal matrix_movie.mp4 file.
7) Once launched run python3 peer.py to launch an instance of a peer from the local terminal
8) When prompted to enter your files, enter ips.txt, bruh.mp7 since our system only takes files found in the peers local directory, ips.txt will register while bruh.mp7 wont
9) To start, let's do a simple search for data1.txt which will be a short text file present on the first peer of the swarm we launched. To do this enter 1 into the command line and then enter data1.txt to search for this file name, note the single ip that shows up
10) Next, prompt with 1 to search for a specific file being shared on the system, and In this case we will want to ask for matrix_movie.mp4, notice that 5 ips will show up, these are the peers we launched that all have the file
11) Once we have seen that these peers have this, we will hit 2 to initiate a download and give it the accurate file name: matrix_movie.mp4. This will then download from each of our five peers chunking the file.
12) Then choosing whether or not this file is what we expected we can choose to report malicious behavior or not.
13) Now that the download has finished this file should be shared from our peer as well so to ensure this works properly we want to search (1) again for matrix_movie.mp4 and we should see that there are now 6 ips sharing the file.
14) Next we want to ensure our overwrite is working as intended so we will initiate another download of the matrix_movie.mp4 file and when prompted if we would like to overwrite the existing file we will choose no.
15) Now we will test the system when the main fails and the backup needs to recover it. To do this simply go into the main server terminal and press ctrl c to shut it down. Now perform a search for data2.txt and notice that the backup has been called.
16) Now download data2.txt and report the user to the blacklist after download.

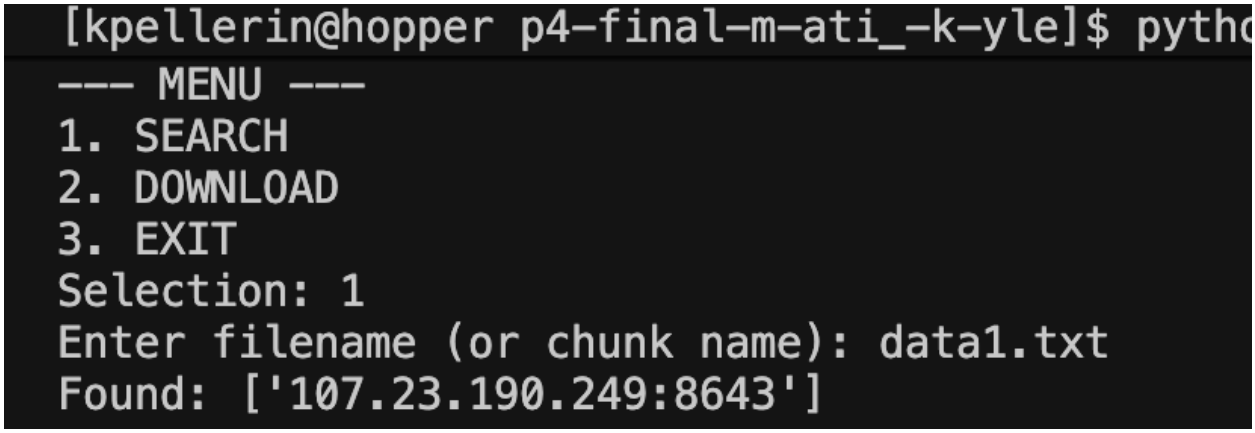
17) Now re launch the main server by entering the command from 5 again, notice that this time it will copy the file list and the blacklisted user over.
18) Lastly, we will unregister our peer by hitting 3 to exit. Go to the main server log to see that the peer has unregistered its files
19) Now shut down the rest of the system by entering ctrl c in the main and backup terminals and then running the kill_swarm.sh

20)



21)

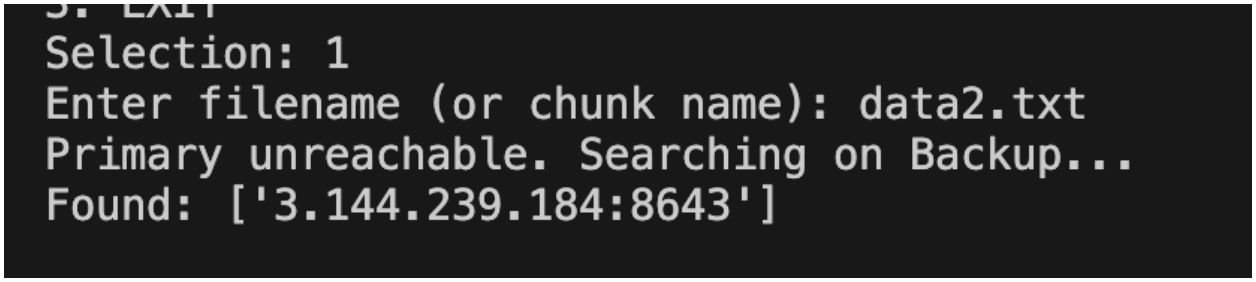


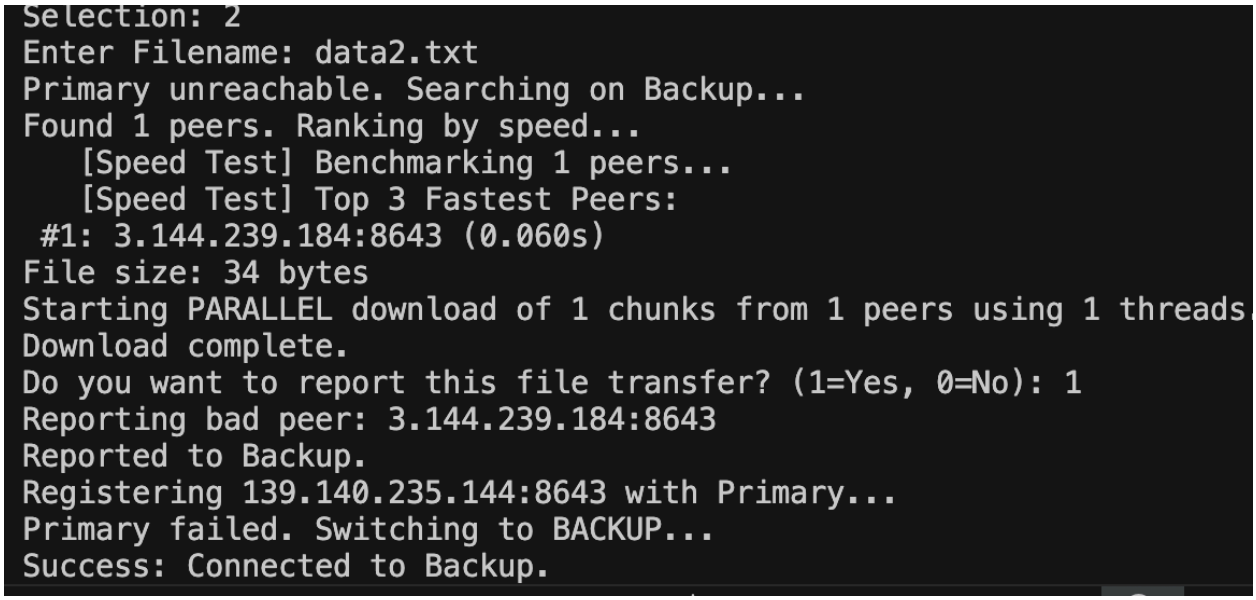


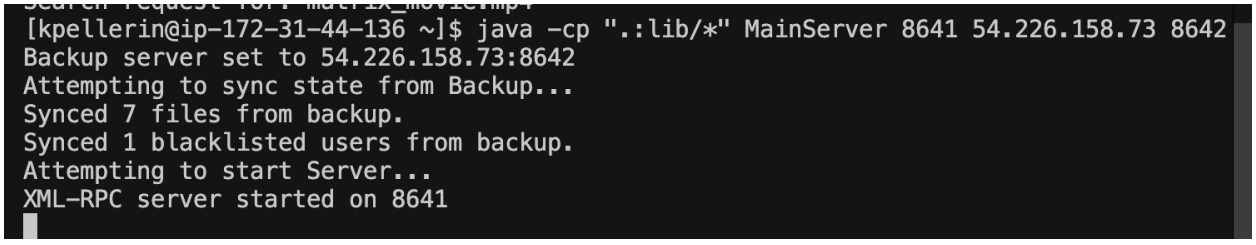
22)
```
Selection: 2
Enter Filename: matrix_movie.mp4
Found 5 peers. Ranking by speed...
   [Speed Test] Benchmarking 5 peers...
   [Speed Test] Top 3 Fastest Peers:
 #1: 107.23.190.249:8643 (0.048s)
 #2: 3.144.239.184:8643 (0.060s)
 #3: 54.151.0.141:8643 (0.163s)
File size: 104857600 bytes
Starting PARALLEL download of 100 chunks from 5 peers using 5 threads...
Download complete.
```

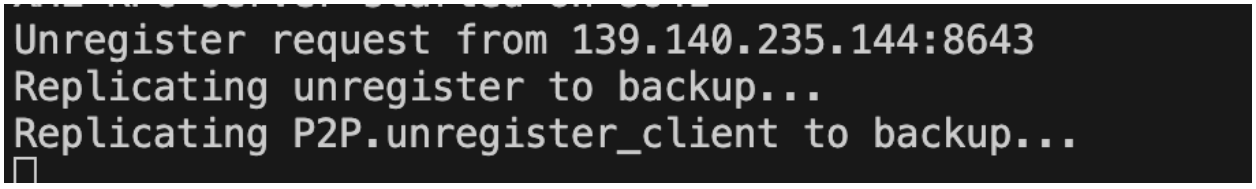
23)
```
Selection: 1
Enter filename (or chunk name): matrix_movie.mp4
Found: ['107.23.190.249:8643', '3.144.239.184:8643', '54.151.0.141:8643', '35.159.30.
5:8643', '35.72.34.224:8643', '139.140.235.144:8643']
```

24)
```
 200 -
   [Speed Test] Top 3 Fastest Peers:
 #1: 139.140.235.144:8643 (0.009s)
 #2: 107.23.190.249:8643 (0.046s)
 #3: 3.144.239.184:8643 (0.067s)
Optimization: Dropping -4 slow peers. Keeping Top 10.
139.140.235.144 - - [18/Dec/2025 12:21:04] "HEAD /download/matrix_movie.mp4 HTTP/1.1"
 200 -
File size: 104857600 bytes
File exists. Overwrite? (1=Yes, 0=No): 0
```

25)

Selection: 1
Enter filename (or chunk name): data2.txt
Primary unreachable. Searching on Backup...
Found: ['3.144.239.184:8643']


Selection: 2
Enter Filename: data2.txt
Primary unreachable. Searching on Backup...
Found 1 peers. Ranking by speed...
[Speed Test] Benchmarking 1 peers...
[Speed Test] Top 3 Fastest Peers:
#1: 3.144.239.184:8643 (0.060s)
File size: 34 bytes
Starting PARALLEL download of 1 chunks from 1 peers using 1 threads
Download complete.
Do you want to report this file transfer? (1=Yes, 0=No): 1
Reporting bad peer: 3.144.239.184:8643
Reported to Backup.
Registering 139.140.235.144:8643 with Primary...
Primary failed. Switching to BACKUP...
Success: Connected to Backup.

26)

[kpellerin@ip-172-31-44-136 ~]$ java -cp ".:lib/*" MainServer 8641 54.226.158.73 8642
Backup server set to 54.226.158.73:8642
Attempting to sync state from Backup...
Synced 7 files from backup.
Synced 1 blacklisted users from backup.
Attempting to start Server...
XML-RPC server started on 8641

27)

Unregister request from 139.140.235.144:8643
Replicating unregister to backup...
Replicating P2P.unregister_client to backup...