Robert Judka
CS550
Programming Assignment 1

Manual

Requirements

This system was developed in C++11 and compiled with the g++ compiler and a Linux subsystem. Running this will not work on a Windows (possibly not MacOS either) subsystem as it does not have the required directory interface.

Build

In ‘src/’, running

```
make all
```

will generate the indexing_server and peer executables. It will also create the peer directories used in this manual and the log directory where the system logs could be found (this may fail if you are not within the ‘src/’ directory).

Execute

To start the indexing server, run

```
./indexing_server
```

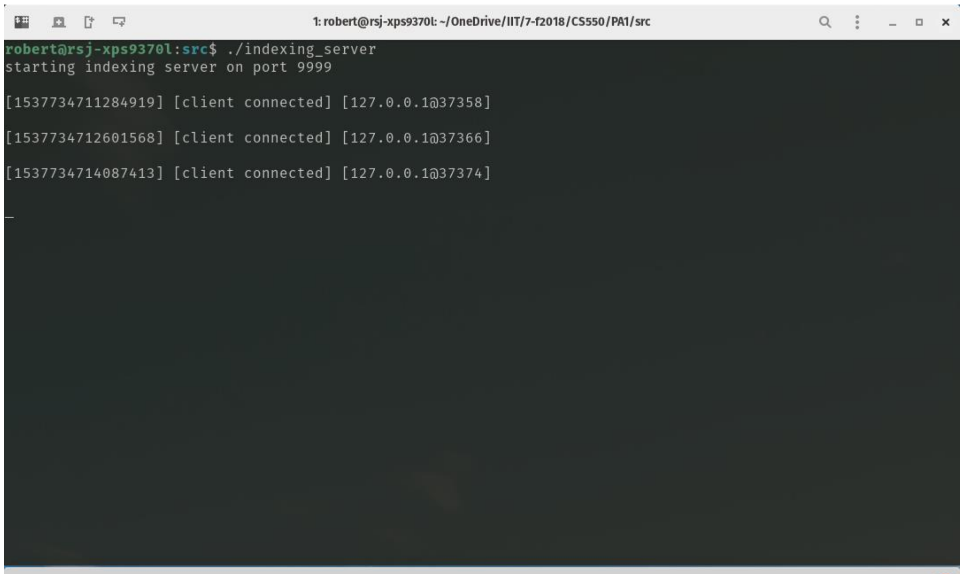
To start a peer, in a separate terminal run

```
./peer {path_to_your_directory}
```

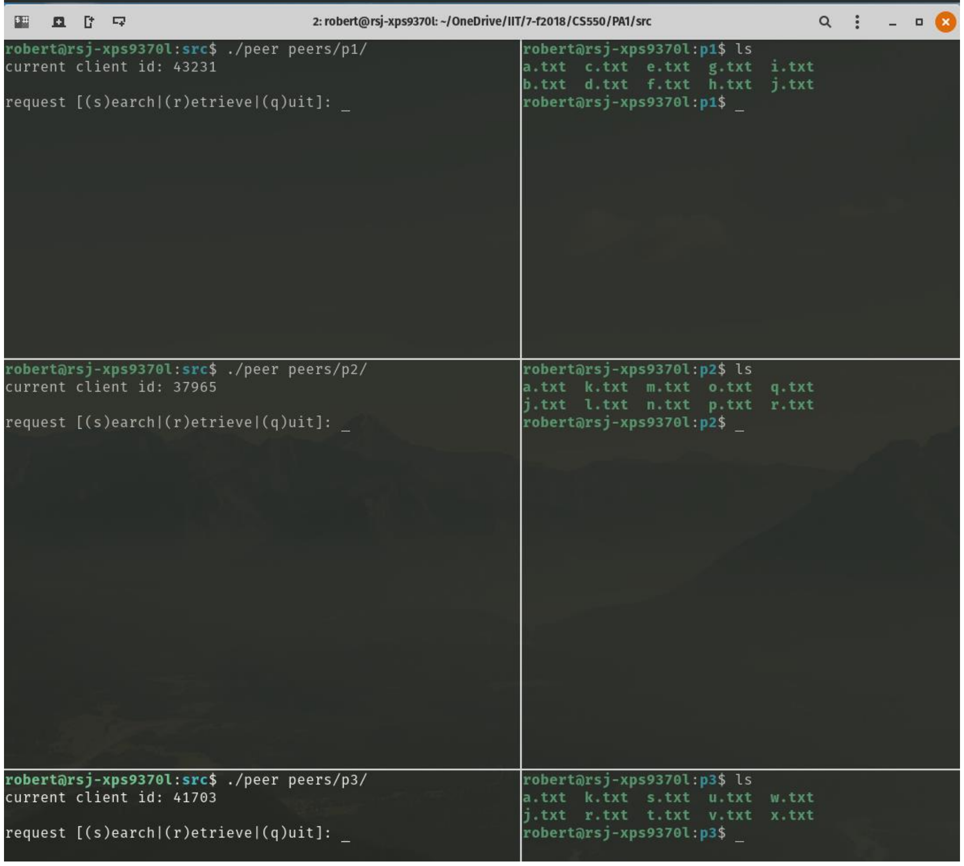
Demo

For convenience, 3 peer directories were prepopulated with files. You can run these 3 peers (in 3 separate terminates) with ‘peers/p1’, ‘peers/p2’, and ‘peers/p3’ as their directories. NOTE these directories will only exist if using the build procedure from above and stay within the ‘src/’ direcotry.

This will the state of your system after executing the indexing_server and 3 peers:


1: robert@rsj-xps9370l: ~/OneDrive/IIT/7-f2018/CS550/PA1/src

```
robert@rsj-xps9370l:src$ ./indexing_server
starting indexing server on port 9999

[1537734711284919] [client connected] [127.0.0.1@37358]

[1537734712601568] [client connected] [127.0.0.1@37366]

[1537734714087413] [client connected] [127.0.0.1@37374]


_
```


2: robert@rsj-xps9370l: ~/OneDrive/IIT/7-f2018/CS550/PA1/src

```
robert@rsj-xps9370l:src$ ./peer peers/p1/
current client id: 43231

request [(s)earch|(r)etrieve|(q)uit]: _
```

```
robert@rsj-xps9370l:p1$ ls
a.txt  c.txt  e.txt  g.txt  i.txt
b.txt  d.txt  f.txt  h.txt  j.txt
robert@rsj-xps9370l:p1$ _
```

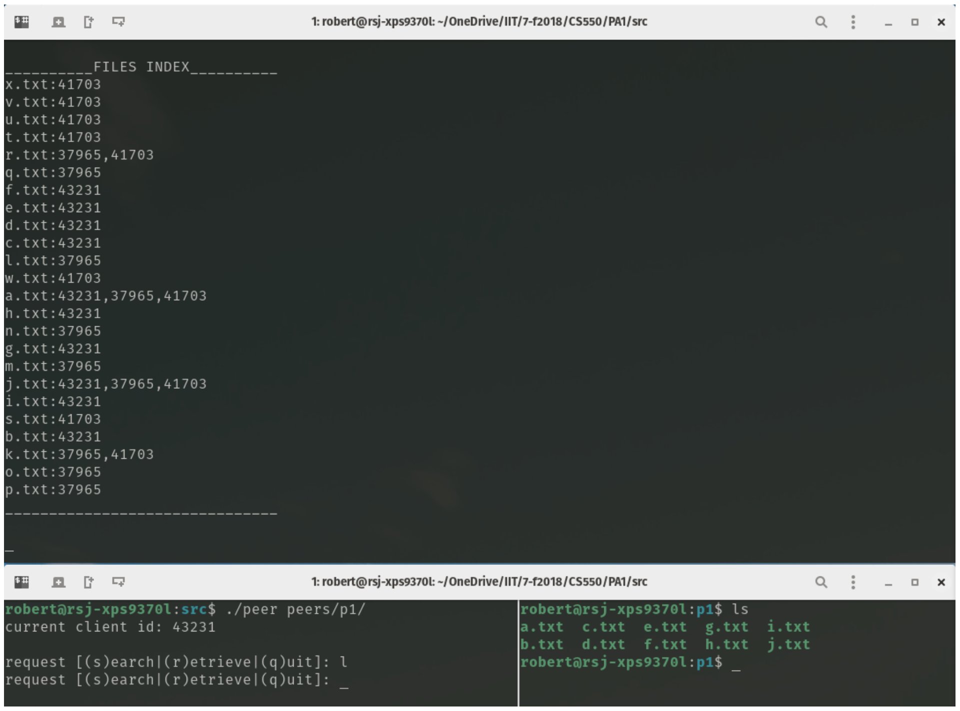
```
robert@rsj-xps9370l:src$ ./peer peers/p2/
current client id: 37965

request [(s)earch|(r)etrieve|(q)uit]: _
```

```
robert@rsj-xps9370l:p2$ ls
a.txt  k.txt  m.txt  o.txt  q.txt
j.txt  l.txt  n.txt  p.txt  r.txt
robert@rsj-xps9370l:p2$ _
```

```
robert@rsj-xps9370l:src$ ./peer peers/p3/
current client id: 41703

request [(s)earch|(r)etrieve|(q)uit]: _
```

```
robert@rsj-xps9370l:p3$ ls
a.txt  k.txt  s.txt  u.txt  w.txt
j.txt  r.txt  t.txt  v.txt  x.txt
robert@rsj-xps9370l:p3$ _
```

To see a list of all registered files, a *hidden* request 'l' could be made through a peer. Here we can see the contents of the *files_index* on the indexing server:

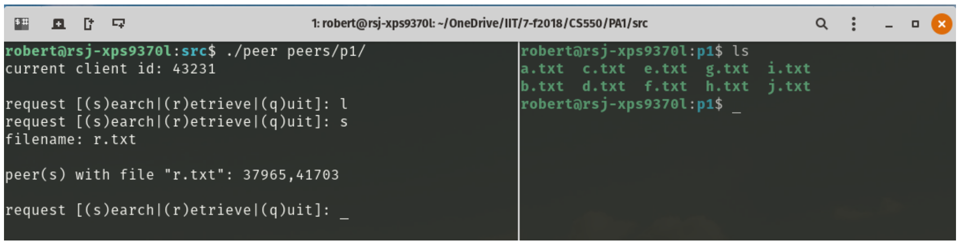


```
1: robert@rsj-xps9370l: ~/OneDrive/IIT/7-f2018/CS550/PA1/src

_________FILES INDEX_________
x.txt:41703
v.txt:41703
u.txt:41703
t.txt:41703
r.txt:37965,41703
q.txt:37965
f.txt:43231
e.txt:43231
d.txt:43231
c.txt:43231
l.txt:37965
w.txt:41703
a.txt:43231,37965,41703
h.txt:43231
n.txt:37965
g.txt:43231
m.txt:37965
j.txt:43231,37965,41703
i.txt:43231
s.txt:41703
b.txt:43231
k.txt:37965,41703
o.txt:37965
p.txt:37965

______________________________

_
```

```
1: robert@rsj-xps9370l: ~/OneDrive/IIT/7-f2018/CS550/PA1/src

robert@rsj-xps9370l:src$ ./peer peers/p1/           robert@rsj-xps9370l:p1$ ls
current client id: 43231                            a.txt  c.txt  e.txt  g.txt  i.txt
                                                    b.txt  d.txt  f.txt  h.txt  j.txt
request [(s)earch|(r)etrieve|(q)uit]: l             robert@rsj-xps9370l:p1$ _
request [(s)earch|(r)etrieve|(q)uit]: _
```

Here we can see all the files registered to the 3 peers, and the files which are shared among the peers.

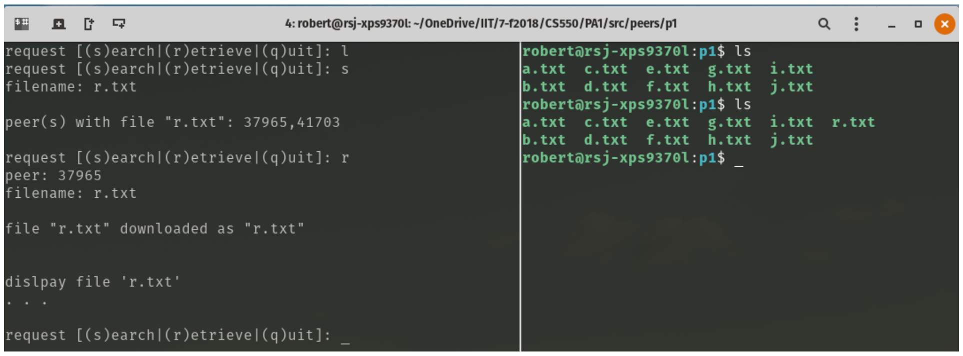
Now, to search for a specific file in the network, we enter 's' for 'search' into the command line and then enter the file we want to search for. In this example, we search for 'r.txt':



```
1: robert@rsj-xps9370l: ~/OneDrive/IIT/7-f2018/CS550/PA1/src

robert@rsj-xps9370l:src$ ./peer peers/p1/           robert@rsj-xps9370l:p1$ ls
current client id: 43231                            a.txt  c.txt  e.txt  g.txt  i.txt
                                                    b.txt  d.txt  f.txt  h.txt  j.txt
request [(s)earch|(r)etrieve|(q)uit]: l             robert@rsj-xps9370l:p1$ _
request [(s)earch|(r)etrieve|(q)uit]: s
filename: r.txt

peer(s) with file "r.txt": 37965,41703

request [(s)earch|(r)etrieve|(q)uit]: _
```

Looking at the first screenshot, we can see that 'r.txt' is owned by both the middle peer [client id 37965] and the bottom peer [client id 41703].

Next, to download a file, we can choose one of the client ids we just learned and the file to download into our directory (assuming we are the top peer [client id 43231]). To do this, we first enter 'r' for 'retrieve' into the command line, then the peer number (in our example we entered '37965'), and finally the file to download, 'r.txt':



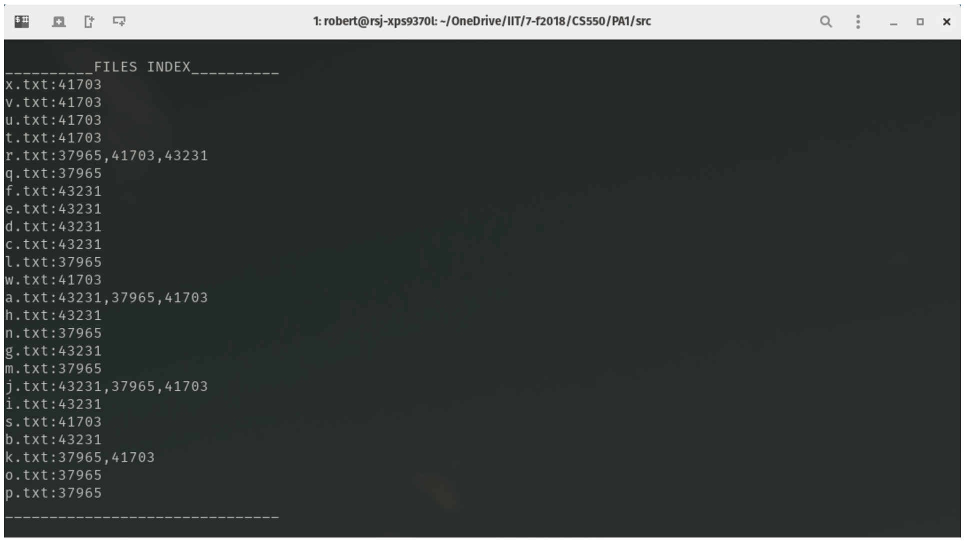
4: robert@rsj-xps9370l: ~/OneDrive/IIT/7-f2018/CS550/PA1/src/peers/p1

```
request [(s)earch|(r)etrieve|(q)uit]: l
request [(s)earch|(r)etrieve|(q)uit]: s
filename: r.txt

peer(s) with file "r.txt": 37965,41703

request [(s)earch|(r)etrieve|(q)uit]: r
peer: 37965
filename: r.txt

file "r.txt" downloaded as "r.txt"


dislpay file 'r.txt'
. . .

request [(s)earch|(r)etrieve|(q)uit]: _
```

```
robert@rsj-xps9370l:p1$ ls
a.txt  c.txt  e.txt  g.txt  i.txt
b.txt  d.txt  f.txt  h.txt  j.txt
robert@rsj-xps9370l:p1$ ls
a.txt  c.txt  e.txt  g.txt  i.txt  r.txt
b.txt  d.txt  f.txt  h.txt  j.txt
robert@rsj-xps9370l:p1$ _
```

We see we have gotten the message saying 'r.txt' was successfully downloaded and attempted to *display* the file ('. . .' symbolizes the content of the file). We can also see on the right side in the terminal pointing to our directory, that by running the 'ls' command, 'r.txt' is now within our directory.

We can also see, by entering 'l' again into the command line, that the *files_index* has been updated to include our new file 'r.txt':

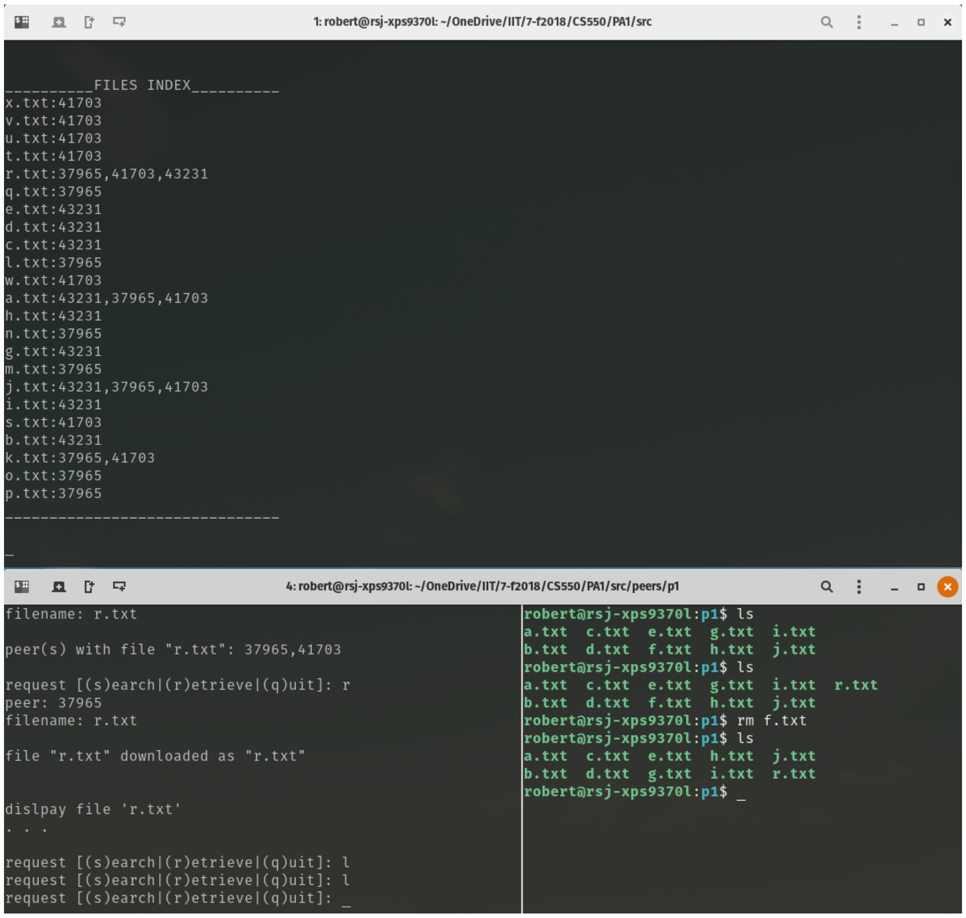


1: robert@rsj-xps9370l: ~/OneDrive/IIT/7-f2018/CS550/PA1/src

```
__________FILES INDEX__________
x.txt:41703
v.txt:41703
u.txt:41703
t.txt:41703
r.txt:37965,41703,43231
q.txt:37965
f.txt:43231
e.txt:43231
d.txt:43231
c.txt:43231
l.txt:37965
w.txt:41703
a.txt:43231,37965,41703
h.txt:43231
n.txt:37965
g.txt:43231
m.txt:37965
j.txt:43231,37965,41703
i.txt:43231
s.txt:41703
b.txt:43231
k.txt:37965,41703
o.txt:37965
p.txt:37965

_______________________________
```

In the indexing server, we can see our client id (assuming we are the top peer [client id 43231]) was added as an owner of 'r.txt'.

Now say we don't want the file 'f.txt' in our directory anymore. After running the 'rm f.txt' command on the right side in the terminal pointing to our directory, and entering 'l' again into the command line, that *files_index* no longer contains our file 'f.txt':



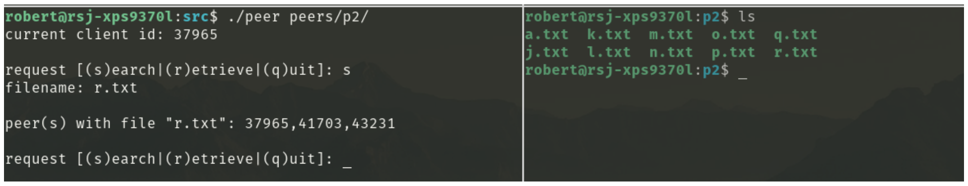
```
1: robert@rsj-xps9370l: ~/OneDrive/IIT/7-f2018/CS550/PA1/src

__________FILES INDEX__________
x.txt:41703
v.txt:41703
u.txt:41703
t.txt:41703
r.txt:37965,41703,43231
q.txt:37965
e.txt:43231
d.txt:43231
c.txt:43231
l.txt:37965
w.txt:41703
a.txt:43231,37965,41703
h.txt:43231
n.txt:37965
g.txt:43231
m.txt:37965
j.txt:43231,37965,41703
i.txt:43231
s.txt:41703
b.txt:43231
k.txt:37965,41703
o.txt:37965
p.txt:37965
-------------------------------

_
```

```
4: robert@rsj-xps9370l: ~/OneDrive/IIT/7-f2018/CS550/PA1/src/peers/p1

filename: r.txt

peer(s) with file "r.txt": 37965,41703

request [(s)earch|(r)etrieve|(q)uit]: r
peer: 37965
filename: r.txt

file "r.txt" downloaded as "r.txt"


dislpay file 'r.txt'
. . .

request [(s)earch|(r)etrieve|(q)uit]: l
request [(s)earch|(r)etrieve|(q)uit]: l
request [(s)earch|(r)etrieve|(q)uit]: _
```

```
robert@rsj-xps9370l:p1$ ls
a.txt  c.txt  e.txt  g.txt  i.txt
b.txt  d.txt  f.txt  h.txt  j.txt
robert@rsj-xps9370l:p1$ ls
a.txt  c.txt  e.txt  g.txt  i.txt  r.txt
b.txt  d.txt  f.txt  h.txt  j.txt
robert@rsj-xps9370l:p1$ rm f.txt
robert@rsj-xps9370l:p1$ ls
a.txt  c.txt  e.txt  h.txt  j.txt
b.txt  d.txt  g.txt  i.txt  r.txt
robert@rsj-xps9370l:p1$ _
```

Here, we can see that 'f.txt' no longer exists in our directory and that change was updated in the *files_index* in the indexing server.

To check that other peers also function properly, we will move to the middle peer [client id 37965], and enter 's' into the command line to search for the file 'r.txt':

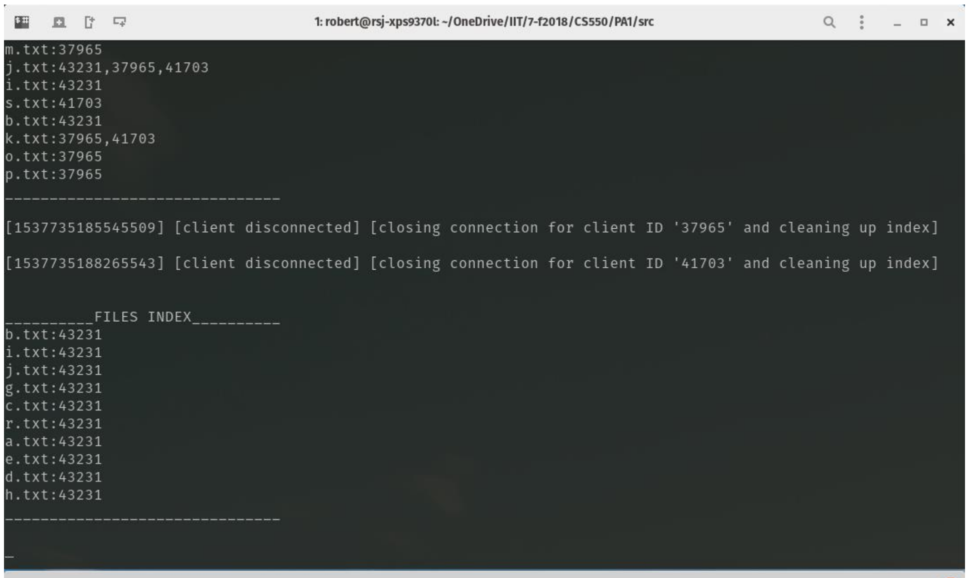


```
robert@rsj-xps9370l:src$ ./peer peers/p2/
current client id: 37965

request [(s)earch|(r)etrieve|(q)uit]: s
filename: r.txt

peer(s) with file "r.txt": 37965,41703,43231

request [(s)earch|(r)etrieve|(q)uit]: _
```

```
robert@rsj-xps9370l:p2$ ls
a.txt  k.txt  m.txt  o.txt  q.txt
j.txt  l.txt  n.txt  p.txt  r.txt
robert@rsj-xps9370l:p2$ _
```

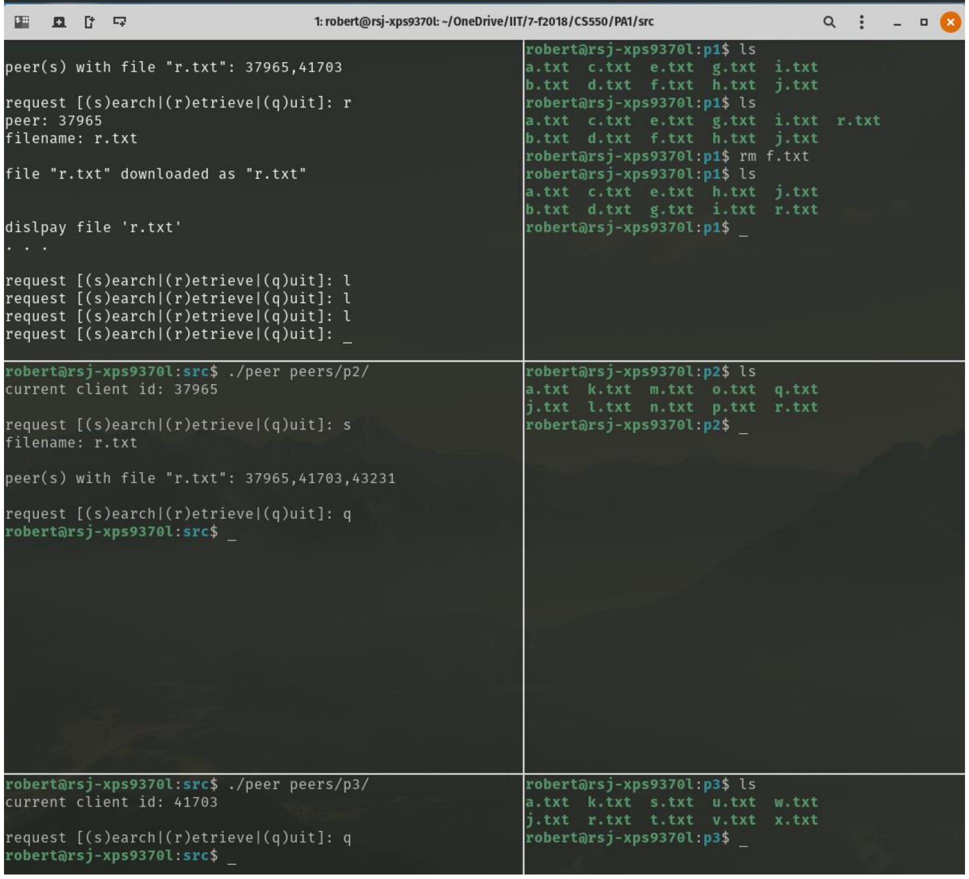
We see again that 'r.txt' is owner by all 3 peers.

Finally, we want to remove the middle peer [client id 37965] and the bottom peer [client id 41703] by entering ‘q’ for quit into both of their command lines. Then, to verify both those peers were removed from the network, we enter ‘l’ into the command line for the top peer [client id 43231]:



1: robert@rsj-xps9370l: ~/OneDrive/IIT/7-f2018/CS550/PA1/src

```
m.txt:37965
j.txt:43231,37965,41703
i.txt:43231
s.txt:41703
b.txt:43231
k.txt:37965,41703
o.txt:37965
p.txt:37965
--------------------------------

[1537735185545509] [client disconnected] [closing connection for client ID '37965' and cleaning up index]

[1537735188265543] [client disconnected] [closing connection for client ID '41703' and cleaning up index]


__________FILES INDEX__________
b.txt:43231
i.txt:43231
j.txt:43231
g.txt:43231
c.txt:43231
r.txt:43231
a.txt:43231
e.txt:43231
d.txt:43231
h.txt:43231
--------------------------------

_
```



1: robert@rsj-xps9370l: ~/OneDrive/IIT/7-f2018/CS550/PA1/src

```
peer(s) with file "r.txt": 37965,41703

request [(s)earch|(r)etrieve|(q)uit]: r
peer: 37965
filename: r.txt

file "r.txt" downloaded as "r.txt"


dislpay file 'r.txt'
. . .

request [(s)earch|(r)etrieve|(q)uit]: l
request [(s)earch|(r)etrieve|(q)uit]: l
request [(s)earch|(r)etrieve|(q)uit]: l
request [(s)earch|(r)etrieve|(q)uit]: _
```

```
robert@rsj-xps9370l:p1$ ls
a.txt  c.txt  e.txt  g.txt  i.txt
b.txt  d.txt  f.txt  h.txt  j.txt
robert@rsj-xps9370l:p1$ ls
a.txt  c.txt  e.txt  g.txt  i.txt  r.txt
b.txt  d.txt  f.txt  h.txt  j.txt
robert@rsj-xps9370l:p1$ rm f.txt
robert@rsj-xps9370l:p1$ ls
a.txt  c.txt  e.txt  h.txt  j.txt
b.txt  d.txt  g.txt  i.txt  r.txt
robert@rsj-xps9370l:p1$ _
```

```
robert@rsj-xps9370l:src$ ./peer peers/p2/
current client id: 37965

request [(s)earch|(r)etrieve|(q)uit]: s
filename: r.txt

peer(s) with file "r.txt": 37965,41703,43231

request [(s)earch|(r)etrieve|(q)uit]: q
robert@rsj-xps9370l:src$ _
```

```
robert@rsj-xps9370l:p2$ ls
a.txt  k.txt  m.txt  o.txt  q.txt
j.txt  l.txt  n.txt  p.txt  r.txt
robert@rsj-xps9370l:p2$ _
```

```
robert@rsj-xps9370l:src$ ./peer peers/p3/
current client id: 41703

request [(s)earch|(r)etrieve|(q)uit]: q
robert@rsj-xps9370l:src$ _
```

```
robert@rsj-xps9370l:p3$ ls
a.txt  k.txt  s.txt  u.txt  w.txt
j.txt  r.txt  t.txt  v.txt  x.txt
robert@rsj-xps9370l:p3$ _
```

In the indexing server, we can see that the only files in the *files_index* are those which are owned by the top peer [client id 43231].

*more detailed outputs can be viewed in the ‘logs/’ directory