

Operating Systems(PG)
Monsoon 2016
Assignment 2
Deadline: Tuesday 6 September 2016 11:55 PM

Date: 28 August 2016

Objective : Create LAN file sharing system using socket programming

Tasks of this assignment is mainly divided into two parts.

PART 1:

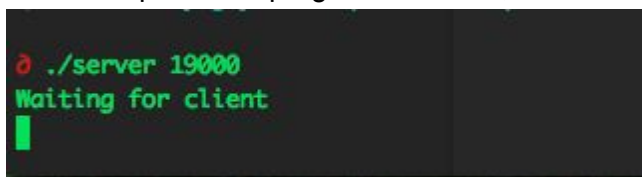
Write program to create a central repo server which would handle the access to central share repository. Central repository is a simple file which maintains a map of file path and machine ip, from which the file can be downloaded

Program should listen and respond to following operations

- Search for a file in repo, and return the mirrors
- Add a new share entry in the repo file as requested by a client
- You are free to design your own wire protocol to distinguish among requests.
- Maintain a separate log file which logs all the incoming and outgoing events at the server. Name the file "repo.log" and place it in the same directory as the executable.
- Following is the logging format. Log all events on a new line.
 - <timestamp>: <event-description>
 - E.gs:
 - 28-08-2016 16-30-27: Search request from 10.1.1.2
 - 28-08-2016 16-30-27: Search response send to 10.1.1.2
 - 28-08-2016 16-30-47: Share request from 10.1.1.2
 - 28-08-2016 16-30-47: Share ack sent to 10.1.1.2

Note: Repo server only handles operation on the repo.txt file. Such as add a new entry or retrieve desired mirrors for a search query. It does not act as a file download server.

Run the repo server program as below, where 19000 is the port on which the server listens



```
➤ ./server 19000
Waiting for client
```

PART 2:

Write a simple client program(which doubles up as a server for other nodes to download file) which will provide a menu driven interface to do the following:

- Search for a file to download
- Share a file on central repository
- Also this same client, **should fork a new process** to run a file download server parallelly. Again you need to design your own wire protocol to understand incoming requests and respond accordingly.

- Maintain a separate log file which logs all the incoming and outgoing events at the server. Name the file “client.log” and place it in the same directory as the executable.
- Following is the logging format. Log all events on a new line.
 - <timestamp>: <event-description>
 - E.gs:
 - 28-08-2016 16-30-27: Download request from 10.1.1.2
 - 28-08-2016 16-30-27: File sent to 10.1.1.2

Search feature will allow user to add keywords and that would be queried to central repository via sockets. The result returned would be a list of options(mirrors) from which the file can be downloaded. Once a mirror is selected the file must be downloaded from the selected mirror.

Share a file, is a simple feature where user gives in a filepath to share. The same is queried to repo server to append a new entry in remote repository.

Run the client program as below, where

- Argument 1 : ip of the machine where repo server is running. (localhost or 127.0.0.1 for local machine)
- Argument 2 : port number on which the repo server is hosted

```

server
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
}
cas
}
}
close(socke
}
int main(int ar
{
int fd = fo

```

```

./client localhost 19000
Select:
1. Search
2. Share
3. Exit
>> 1
Type string to search: test
Select a mirror :
1. test_client.txt 127.0.0.1
2. test.jpg 127.0.0.1
>> 2
Download complete!
Select:
1. Search
2. Share
3. Exit
>> 2
Path : ./test.jpg
File Shared Successfully on repo!
Select:
1. Search
2. Share
3. Exit
>> 3

```

Test cases to be handled

- Host unreachable, incase the repo server or client is not reachable.
- Repository list can be outdated for certain records. I.e file may not be present at the specified location. Such cases need to be handled with appropriate error
- Repository listed mirror may be offline(client program on that machine may not be online). Such cases need to be handled with appropriate error
- Socket binding errors need to be handled with appropriate error message.

Pointers :

- We recommend you to learn and use Makefiles (not mandatory).
- Learn how to code, a simple socket connection in c++. Test by sending and receiving messages between client and server.
- Each operation in this assignment is mainly to establish a socket connection and send data through it.
- Read more about fork() call. Question yourself why do we need it in case of client program. Can we do without it?
- Write sample codes to append data to files in c++. This will help in updating the repository on repo server. Also this will aid in implementing log files.
- Wire protocol mentioned here, is nothing but the way you would distinguish different incoming requests. For e.g repo server would listen for two kinds of requests namely 'Search' and 'Share'. One way to do this is to send messages as follows:
 - search#@#music
 - share#@#./test.txt
 - Here, '@#' is a delimiter, now at the receiving end you can split the message using '@#' as delimiter and distinguish among different types of messages and perform tasks accordingly.

How will this assignment be evaluated

- Test 1 : Search, Share and Download features would be tested on local machine. Both repo server and client running on the same machine.
- Test 2 : Search, Share and Download features would be tested on local machine. Repo server and client running on the different machine.

General Guidelines :

- You can use STLs in CPP.
- Indent and comment the code properly.
- Your name and roll number should be included as comments at the beginning of the code
- Due credit will be given to modularity of code.
- ZERO tolerance towards any kind of code plagiarism.
- Strictly follow the upload format and deadlines. All invalid submissions will not be considered for evaluation.
- Make sure you do not upload any executables.
- Handle error cases wherever required.

Upload format :

- Create a folder named your roll number(20XXXXXXX_Assignment2). Inside that. In each folder, place your '.c' or '.cpp' files, Makefile(if any).
- Create a tar.gz of the above folder(20XXXXXXX_Assignment2) named "20XXXXXXX_Assignment2.tar.gz" and upload it. Example:
2015123433_Assignment1.tar.gz