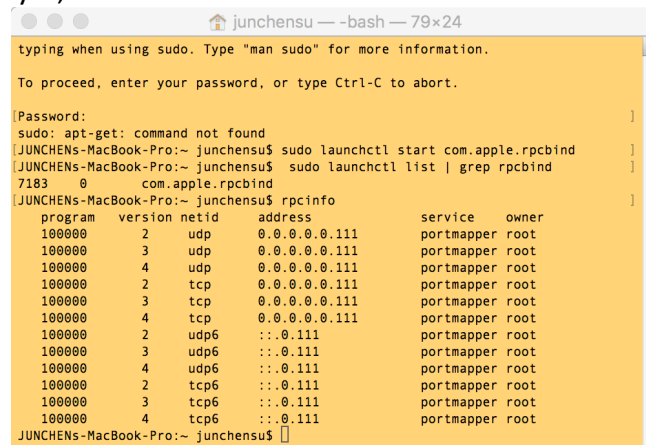


In this assignment, we are required to use RPC, so we need to use RPC tool rpcgen, in my mac OS, it was not installed yet, so I installed RPCbind in order to run related code. See Figure 1.



```

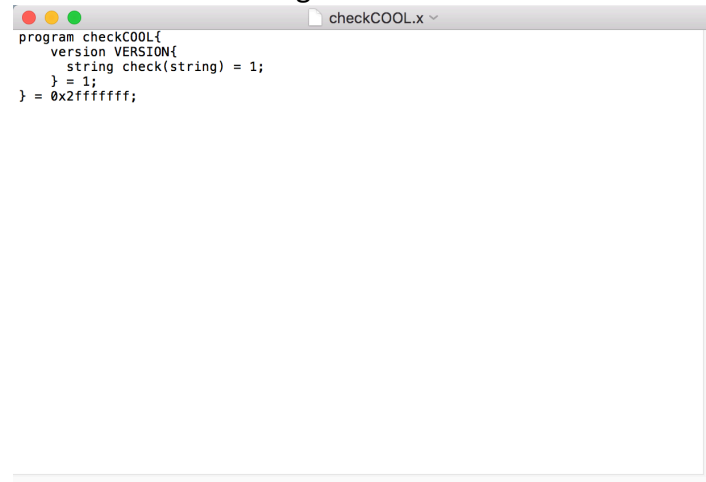
junchensu ~ -bash — 79x24
typing when using sudo. Type "man sudo" for more information.
To proceed, enter your password, or type Ctrl-C to abort.

[Password:
sudo: apt-get: command not found
JUNCHENS-MacBook-Pro:~ junchensu$ sudo launchctl start com.apple.rpcbind
JUNCHENS-MacBook-Pro:~ junchensu$ sudo launchctl list | grep rpcbind
7183 0 com.apple.rpcbind
JUNCHENS-MacBook-Pro:~ junchensu$ rpcinfo
program version netid address service owner
100000 2 udp 0.0.0.0.0.111 portmapper root
100000 3 udp 0.0.0.0.0.111 portmapper root
100000 4 udp 0.0.0.0.0.111 portmapper root
100000 2 tcp 0.0.0.0.0.111 portmapper root
100000 3 tcp 0.0.0.0.0.111 portmapper root
100000 4 tcp 0.0.0.0.0.111 portmapper root
100000 2 udp6 ::0.111 portmapper root
100000 3 udp6 ::0.111 portmapper root
100000 4 udp6 ::0.111 portmapper root
100000 2 tcp6 ::0.111 portmapper root
100000 3 tcp6 ::0.111 portmapper root
100000 4 tcp6 ::0.111 portmapper root
JUNCHENS-MacBook-Pro:~ junchensu$

```

Figure.1

Then we need to write a .x source code to generated several RPC files. See Figure 2.



```

checkCOOL.x
program checkCOOL{
    version VERSION{
        string check(string) = 1;
    } = 1;
} = 0x2fffffff;

```

Figure 2

In this file, we have to name our program name, as well as the function name which is inside the program. And the number in this bottom can be random number. After this, we enter this directory and run `rpc -a -C checkCOOL.x` in terminal. Then this operation will generate 6 files for us. And next we need to modify the server file and client file to achieve our functionality.

In the client side, fist we need to create client stub using:

```
clnt = clnt_create (host, checkCOOL, VERSION, "udp");
```

Report error if any occur. And then we can implement our function like Assigement3. And we set the IP 127.0.0.1 which is a common IP. Once client has something to send, it will send to client stub and the stub will encapsulate it and send to sever.

In the server side, we just simply implement the method to compare “COOL”. When the server stub found any request, it will decode it and send to server. And server will call the method to get the result and send it to stub. Finally stub return it back to client.

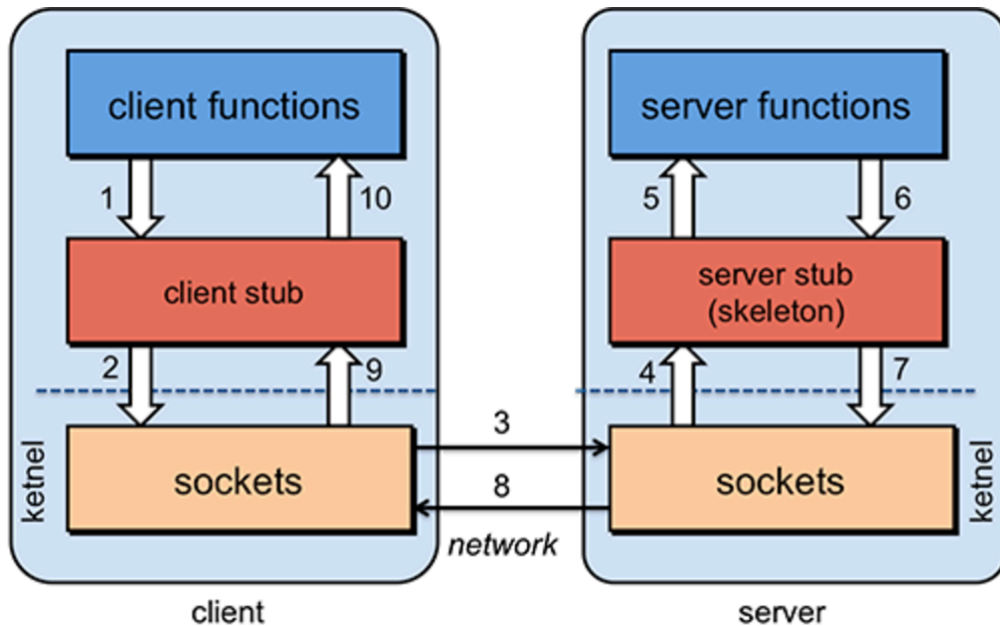


Figure 3

Finally we use “make -f Makefile.checkCOOL” to compile the codes. See the result in Figure 4.

```
Please enter a alpha numeric string: sdad
no cool included
Please enter a alpha numeric string: C00L21313
Please enter a alpha numeric string: SJCSB
no cool included
Please enter a alpha numeric string: C00L12111
Please enter a alpha numeric string: 
The input is C00L21313, it has 7 digits.
The input is C00L12111, it has 7 digits.
^CTotal input lines:2
Total input digist:14
```

Figure.4

