**Cpre 530 - Program 5**

**Fall 2011**

**Suganya Baskaran**

#include <netinet/in.h>

#include <netdb.h>

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/time.h>

#include <sys/socket.h>

struct sockaddr\_in s\_in, temp, from\_addr;

int from\_len;

extern int errno;

int debug = 0;

char \* File\_read(char \*filename);

int main(int argc, char \*\*argv){

struct timeval timeout;

register int n;

u\_short len;

char \*cp;

struct hostent \*h\_name;

struct servent \*s\_name;

char senderEmail[70];

char recipient[70];

int numTimeOuts = 0;

int i, retry, resplen, done = 0;

int dsmask, flags, sockFD, file\_status =0, recip\_status =0,

sender\_status =0;

char buf[100],answer[4048],user[100],filename[50],sender\_id[100];

opterr = 0;

sockFD = -1;

h\_name = gethostbyname("bones.ee.iastate.edu");

s\_name = getservbyname("smtp", "tcp");

s\_in.sin\_port = s\_name->s\_port;

s\_in.sin\_family = AF\_INET;

s\_in.sin\_addr.s\_addr = \*(u\_long \*)h\_name->h\_addr;

while ((i = getopt(argc, argv, "sudf")) != -1)

{

switch (i)

{

case 'u':

strcpy(user ,argv[optind]);

recip\_status = 1;

break;

case 'f':

strcpy(filename, argv[optind]);

file\_status = 1;

break;

case 's':

strcpy(sender\_id ,argv[optind]);

sender\_status = 1;

break;

case 'd':

debug = 1;

flags = 4;

break;

case '?':

default:

done = 1;

flags = 0;

break;

}

if (done) break;

}

printf("port = %d -- %s\n",ntohs(s\_in.sin\_port),inet\_ntoa(s\_in.sin\_addr));

sockFD = socket(AF\_INET, SOCK\_STREAM, 0);

if (connect(sockFD, (struct sockaddr \* )&s\_in, sizeof(s\_in)) < 0) {

perror("connect request");

(void) close(sockFD);

exit(1);

}

if (getline(sockFD, answer) == 0){

printf("No Reply\n");

exit(0);

}

if (answer[0] == '2') printf("Server Reply OK\n");

printf("\n[%s]\n", answer);

strcpy(buf," unknown.iastate.edu\n");

printf("Client Request: %s",buf);

if (send(sockFD, buf, strlen(buf),0) != strlen(buf)) {

perror("send request");

(void) close(sockFD);

exit(1);

}

if (getline(sockFD, answer) == 0){

printf("No Reply\n");

exit(0);

}

printf("Server Reply: %s\n\n",answer);

if (answer[0] != '2'){

printf("Error\n");

exit(0);

}

if(sender\_status == 1)

{

strcpy(senderEmail,"MAIL FROM: ");

strcat(senderEmail,sender\_id);

strcat(senderEmail,"\n");

strcpy(buf,senderEmail);

}

else

{

strcpy(buf,"MAIL FROM: somebody@iastate.edu\n");

}

printf("Client Says: %s",buf);

if (send(sockFD, buf, strlen(buf),0) != strlen(buf)) {

perror("send request");

(void) close(sockFD);

exit(1);

}

if (getline(sockFD, answer) == 0){

printf("No Reply\n");

exit(0);

}

printf("Server Says: %s\n\n",answer);

if (answer[0] != '2')

{

printf("Error\n");

exit(0);

}

if(recip\_status == 1)

{

strcpy(recipient,"RCPT TO:");

strcat(recipient,user);

strcat(recipient,"@bones.ee.iastate.edu\n");

strcpy(buf,recipient);

}

else

{

strcpy(buf,"RCPT TO: suganya@bones.ee.iastate.edu\n");

}

printf("Client Request: %s",buf);

if (send(sockFD, buf, strlen(buf),0) != strlen(buf)) {

perror("send request");

(void) close(sockFD);

exit(1);

}

if (getline(sockFD, answer) == 0){

printf("No Reply\n");

exit(0);

}

printf("Server Says: %s\n\n",answer);

strcpy(buf,"DATA\n");

printf("Client Says: %s",buf);

if (send(sockFD, buf, strlen(buf),0) != strlen(buf)) {

perror("send request");

(void) close(sockFD);

exit(1);

}

if (getline(sockFD, answer) == 0){

printf("No Reply\n");

exit(0);

}

printf("Server Says: %s\n\n",answer);

if(file\_status == 1 )

{

strcpy(buf,File\_read(filename));

strcat(buf,"\n.\n");

}

else

{

strcpy(buf,": This is the content of the mail.\n.\n");

}

printf("Client Request: %s",buf);

if (send(sockFD, buf, strlen(buf),0) != strlen(buf)) {

perror("send request");

(void) close(sockFD);

exit(1);

}

if (getline(sockFD, answer) == 0){

printf("No Reply\n");

exit(0);

}

printf("Server Says: %s\n\n",answer);

strcpy(buf,"QUIT\n");

printf("Client Says: %s",buf);

if (send(sockFD, buf, strlen(buf),0) != strlen(buf)) {

perror("send request");

(void) close(sockFD);

exit(1);

}

if (getline(sockFD, answer) == 0){

printf("No Reply\n");

exit(0);

}

printf("Server Says: %s\n\n",answer);

(void) close(sockFD);

exit(1);

}

int getline(int fd, char \*answer)

{

char \*cp = answer;

struct timeval timeout;

int dsmask, reply, done = 0;

char buf[100];

int n = 0;

answer[0] = 0;

while (1)

{

timeout.tv\_sec = 4;

timeout.tv\_usec = 0;

dsmask = 1 << fd;

n = select(fd+1, &dsmask, 0, 0, &timeout);

if (n < 0) {

perror("spam select error");

return 0;

}

if (n == 0) {

printf("spam: mask = %d after select call\n",dsmask);

return 0;

}

if ((n = recv(fd, buf, 100, 0)) < 0){

perror("recv");

(void) close(fd);

return 0;

}

if ((buf[n-1] == '\n') || (buf[n-1] == '\r')) done = 1;

buf[n-1]=0;

if (debug) printf("<%s>\n", buf);

strcat(answer, buf);

if (done) return 1;

}

}

char \* File\_read(char \*filename)

{

int i = 0;

char buf[10000];

FILE \*file;

char c;

file = fopen(filename,"r");

if(file == NULL)

{

printf("Error!\n");

}

else

{

while((c = fgetc(file)) != EOF)

{

buf[i] = putchar(c);

i++;

}

fclose(file);

}

return buf;

}