**Team5Lab2Contents.docx**

**Q1:**

**Server\_Iterative\_tcp.c**

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <unistd.h>

#include <arpa/inet.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <math.h>

#include <ctype.h>

int main(){

//create a socket

int serv\_sock = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

//assign the address and port

struct sockaddr\_in serv\_addr;

memset(&serv\_addr, 0, sizeof(serv\_addr)); //initial server address

serv\_addr.sin\_family = AF\_INET; //use IPV4

serv\_addr.sin\_addr.s\_addr = inet\_addr("127.0.0.1"); //use local host

serv\_addr.sin\_port = htons(2222); //port is 2222

bind(serv\_sock, (struct sockaddr\*)&serv\_addr, sizeof(serv\_addr));

//listen the port

listen(serv\_sock, 20);

char fname[50];

//Iteratively handle each requet

while(1) {

printf("Waiting...\n");

//accept the request from client

struct sockaddr\_in clnt\_addr;

socklen\_t clnt\_addr\_size = sizeof(clnt\_addr);

int connfd = accept(serv\_sock, (struct sockaddr\*)&clnt\_addr, &clnt\_addr\_size);

read(connfd, fname,256);

printf("Message form client: %s\n", fname);

FILE \*fp = fopen(fname,"rb");

if (fp==NULL) {

printf("There is no file named: %s\n", fname);

} else {

//Read data from file and send it

while(1) {

unsigned char buff[1024]={0};

int nread = fread(buff,1,1024,fp);

//If read was success, send data

if(nread > 0) {

write(connfd, buff, nread);

}

if (nread < 1024) {

if (feof(fp)) {

printf("End of file\n");

printf("File transfer completed for id: %d\n",connfd);

}

if (ferror(fp)) {

printf("Error reading\n");

}

break;

}

}

}

//close the thread

printf("Closing Connection for id: %d\n",connfd);

close(connfd);

sleep(2);

}

}

**server\_MultiProcessing\_tcp.c**

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <errno.h>

#include <string.h>

#include <sys/types.h>

#include <pthread.h>

struct sockaddr\_in c\_addr;

char fname[100];

// send file to client function

int SendFileToClient(int fd)

{

//information from the client

int connfd = fd;

printf("Connection accepted and id: %d\n",connfd);

printf("Connected to Client: %s:%d\n",inet\_ntoa(c\_addr.sin\_addr),ntohs(c\_addr.sin\_port));

//file which requested by client

read(connfd, fname,256);

printf("Message form client: %s\n", fname);

FILE \*fp = fopen(fname,"rb");

if (fp==NULL) {

printf("There is no file named: %s\n", fname);

} else {

//Read data from file and send it

while(1) {

unsigned char buff[1024]={0};

int nread = fread(buff,1,1024,fp);

//If read was success, send data

if(nread > 0) {

write(connfd, buff, nread);

}

if (nread < 1024) {

if (feof(fp)) {

printf("End of file\n");

printf("File transfer completed for id: %d\n",connfd);

}

if (ferror(fp)) {

printf("Error reading\n");

}

break;

}

}

}

//close the thread

printf("Closing Connection for id: %d\n",connfd);

close(connfd);

shutdown(connfd,SHUT\_WR);

sleep(2);

return 0;

}

int main() {

int connfd = 0;

struct sockaddr\_in serv\_addr;

int listenfd = 0,ret;

size\_t clen=0;

//server's information

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

if(listenfd<0)

{

printf("Error in socket creation\n");

exit(2);

}

serv\_addr.sin\_family = AF\_INET;

serv\_addr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

serv\_addr.sin\_port = htons(2222);

ret=bind(listenfd, (struct sockaddr\*)&serv\_addr,sizeof(serv\_addr));

if(ret<0)

{

printf("Error in bind\n");

exit(2);

}

if(listen(listenfd, 10) == -1)

{

printf("Failed to listen\n");

return -1;

}

//listen to the port, create one thread per request

while(1)

{

clen=sizeof(c\_addr);

printf("Waiting...\n");

connfd = accept(listenfd, (struct sockaddr\*)&c\_addr,&clen);

if(connfd<0)

{

printf("Error in accept\n");

continue;

}

switch (fork()) {

case 0:{

/\* child \*/

(void)close(listenfd);

exit(SendFileToClient(connfd));

}

default:{

/\* parent \*/

(void)close(connfd);

break;

}

case -1:{

printf("Error to fork");

}

}

}

return 0;

}

**server\_MultiThreading\_tcp.c**

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <errno.h>

#include <string.h>

#include <sys/types.h>

#include <pthread.h>

struct sockaddr\_in c\_addr;

char fname[100];

// send file to client function

void \*SendFileToClient(void \*arg)

{

//information from the client

int connfd = \*(int\*)arg;

printf("Connection accepted and id: %d\n",connfd);

printf("Connected to Client: %s:%d\n",inet\_ntoa(c\_addr.sin\_addr),ntohs(c\_addr.sin\_port));

//file which requested by client

read(connfd, fname,256);

printf("Message form client: %s\n", fname);

FILE \*fp = fopen(fname,"rb");

if (fp==NULL) {

printf("There is no file named: %s\n", fname);

} else {

//Read data from file and send it

while(1) {

unsigned char buff[1024]={0};

int nread = fread(buff,1,1024,fp);

//If read was success, send data

if(nread > 0) {

write(connfd, buff, nread);

}

if (nread < 1024) {

if (feof(fp)) {

printf("End of file\n");

printf("File transfer completed for id: %d\n",connfd);

}

if (ferror(fp)) {

printf("Error reading\n");

}

break;

}

}

}

//close the thread

printf("Closing Connection for id: %d\n",connfd);

close(connfd);

shutdown(connfd,SHUT\_WR);

sleep(2);

}

int main() {

int connfd = 0,err;

pthread\_t tid;

struct sockaddr\_in serv\_addr;

int listenfd = 0,ret;

size\_t clen=0;

//server's information

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

if(listenfd<0)

{

printf("Error in socket creation\n");

exit(2);

}

serv\_addr.sin\_family = AF\_INET;

serv\_addr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

serv\_addr.sin\_port = htons(2222);

ret=bind(listenfd, (struct sockaddr\*)&serv\_addr,sizeof(serv\_addr));

if(ret<0)

{

printf("Error in bind\n");

exit(2);

}

if(listen(listenfd, 10) == -1)

{

printf("Failed to listen\n");

return -1;

}

//listen to the port, create one thread per request

while(1)

{ printf("Waiting\n");

clen=sizeof(c\_addr);

connfd = accept(listenfd, (struct sockaddr\*)&c\_addr,&clen);

if(connfd<0)

{

printf("Error in accept\n");

continue;

}

err = pthread\_create(&tid, NULL, SendFileToClient, &connfd);

if (err != 0)

{

printf("\ncan't create thread :[%s]", strerror(err));

}

pthread\_join(err,NULL);

}

close(connfd);

return 0;

}

**server\_PreForked\_tcp.c**

a#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <errno.h>

#include <string.h>

#include <sys/types.h>

#include <pthread.h>

struct sockaddr\_in c\_addr;

char fname[100];

int pid; //preprocessing

void reaper(int);

//clean up zombie children

void reaper(int sig) {

int status;

while (wait3(&status, WNOHANG, (struct rusage \*)0) >= 0)

/\* empty \*/;

}

// send file to client function

int SendFileToClient(int fd)

{

//information from the client

int connfd = fd;

printf("Connection accepted and id: %d\n",connfd);

printf("Connected to Client: %s:%d\n",inet\_ntoa(c\_addr.sin\_addr),ntohs(c\_addr.sin\_port));

//file which requested by client

read(connfd, fname,256);

printf("Message form client: %s\n", fname);

FILE \*fp = fopen(fname,"rb");

if (fp==NULL) {

printf("There is no file named: %s\n", fname);

} else {

//Read data from file and send it

while(1) {

unsigned char buff[1024]={0};

int nread = fread(buff,1,1024,fp);

//If read was success, send data

if(nread > 0) {

write(connfd, buff, nread);

}

if (nread < 1024) {

if (feof(fp)) {

printf("End of file\n");

printf("File transfer completed for id: %d\n",connfd);

}

if (ferror(fp)) {

printf("Error reading\n");

}

break;

}

}

}

//close the thread

printf("Closing Connection for id: %d\n",connfd);

close(connfd);

shutdown(connfd,SHUT\_WR);

sleep(2);

return 0;

}

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

if (argc == 2) {

//get the number of pre-allocated children

char proc[50];

strcpy(proc,argv[1]);

int num = atoi(proc);

int connfd = 0;

struct sockaddr\_in serv\_addr;

int listenfd = 0,ret;

size\_t clen=0;

//server's information

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

if(listenfd<0) {

printf("Error in socket creation\n");

exit(2);

}

(void)signal(SIGCHLD, reaper);

serv\_addr.sin\_family = AF\_INET;

serv\_addr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

serv\_addr.sin\_port = htons(2222);

ret=bind(listenfd, (struct sockaddr\*)&serv\_addr,sizeof(serv\_addr));

if(ret<0) {

printf("Error in bind\n");

exit(2);

}

if(listen(listenfd, 10) == -1) {

printf("Failed to listen\n");

return -1;

}

// preforking process

clen = sizeof(c\_addr);

for (int i = 0; i < num; ++i) {

pid = fork();

if (pid == -1) {

printf("Error to fork");

}

if (pid == 0) {

printf("Preforked child %d\n", i + 1);

// slave(client) sock

connfd = accept(listenfd, (struct sockaddr \*)&c\_addr, &clen);

printf("Preforked child %d connection accept.\n", i + 1);

// for child break the for loop

break;

}

}

//child

if (pid == 0) {

(void)close(listenfd);

exit(SendFileToClient(connfd));

}

//parent

while(1) {

clen=sizeof(c\_addr);

printf("Waiting...\n");

connfd = accept(listenfd, (struct sockaddr\*)&c\_addr,&clen);

if(connfd<0)

{

printf("Error in accept\n");

continue;

}

switch (fork()) {

case 0:{

/\* child \*/

(void)close(listenfd);

exit(SendFileToClient(connfd));

}

default:{

/\* parent \*/

(void)close(connfd);

break;

}

case -1:{

printf("Error to fork");

}

}

}

} else {

printf("Please enter one number\n");

}

return 0;

}

**server\_PreThreaded\_tcp.c**

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <errno.h>

#include <string.h>

#include <sys/types.h>

#include <pthread.h>

struct sockaddr\_in c\_addr;

char fname[100];

// send file to client function

void \*SendFileToClient(void \*arg)

{

//information from the client

int connfd = \*(int\*)arg;

printf("Connection accepted and id: %d\n",connfd);

printf("Connected to client: %s:%d\n",inet\_ntoa(c\_addr.sin\_addr),ntohs(c\_addr.sin\_port));

//file which requested by client

read(connfd, fname,256);

printf("Message form client: %s\n", fname);

FILE \*fp = fopen(fname,"rb");

if (fp==NULL) {

printf("There is no file named: %s\n", fname);

} else {

//Read data from file and send it

while(1) {

unsigned char buff[1024]={0};

int nread = fread(buff,1,1024,fp);

//If read was success, send data

if(nread > 0) {

write(connfd, buff, nread);

}

if (nread < 1024) {

if (feof(fp)) {

printf("End of file\n");

printf("File transfer completed for id: %d\n",connfd);

}

if (ferror(fp)) {

printf("Error reading\n");

}

break;

}

}

}

//close the thread

printf("Closing Connection for id: %d\n",connfd);

close(connfd);

shutdown(connfd,SHUT\_WR);

sleep(2);

}

void \*acceptHandleClient(void \*sock)

{

struct sockaddr\_in fsin; // the address of a client

unsigned int alen; // length of client's address

int listenfd = \*(int \*)sock;

int connfd = accept(listenfd, (struct sockaddr \*)&fsin, &alen);

printf("Prethreaded child connection accepted.\n");

int \*new\_sock = malloc(1);

\*new\_sock = connfd;

SendFileToClient(new\_sock);

return NULL;

}

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

if (argc == 2) {

//get the number of pre-allocated children

char proc[50];

strcpy(proc,argv[1]);

int num = atoi(proc);

int connfd = 0,err;

pthread\_t tid;

struct sockaddr\_in serv\_addr;

int listenfd = 0,ret;

size\_t clen=0;

//server's information

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

if(listenfd<0)

{

printf("Error in socket creation\n");

exit(2);

}

serv\_addr.sin\_family = AF\_INET;

serv\_addr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

serv\_addr.sin\_port = htons(2222);

ret=bind(listenfd, (struct sockaddr\*)&serv\_addr,sizeof(serv\_addr));

if(ret<0)

{

printf("Error in bind\n");

exit(2);

}

if(listen(listenfd, 10) == -1)

{

printf("Failed to listen\n");

return -1;

}

// prethreading process

for (int i = 0; i < num; ++i)

{

pthread\_t child;

int \*new\_sock = &listenfd;

if (pthread\_create(&child, NULL, acceptHandleClient, (void \*)new\_sock) < 0)

printf("Could not creat thread");

else

printf("Prethreaded child %d\n", i + 1);

// detach the child, parent don't need to wait the child finishing.

if (pthread\_detach(child) != 0)

printf("Could not detach thread: %s\n");

}

//listen to the port, create one thread per request

while(1)

{

clen=sizeof(c\_addr);

printf("Waiting...\n");

connfd = accept(listenfd, (struct sockaddr\*)&c\_addr,&clen);

if(connfd<0)

{

printf("Error in accept\n");

continue;

}

err = pthread\_create(&tid, NULL, SendFileToClient, &connfd);

if (err != 0)

{

printf("\ncan't create thread :[%s]", strerror(err));

}

pthread\_join(err,NULL);

}

close(connfd);

} else {

printf("Please enter one number\n");

}

return 0;

}

**Q3:**

**client\_tcp.c**

#include <sys/socket.h>

#include <sys/types.h>

#include <netinet/in.h>

#include <netdb.h>

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <unistd.h>

#include <errno.h>

#include <arpa/inet.h>

#include <pthread.h>

int main(int argc, char \*argv[])

{ if (argc == 2) {

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

//sent the request to the assigned host

struct sockaddr\_in serv\_addr;

memset(&serv\_addr, 0, sizeof(serv\_addr)); //initial the address

serv\_addr.sin\_family = AF\_INET; //use IPv4

serv\_addr.sin\_addr.s\_addr = inet\_addr("127.0.0.1"); //use local host

serv\_addr.sin\_port = htons(2222); //port is 2222

connect(sock, (struct sockaddr\*)&serv\_addr, sizeof(serv\_addr));

//send the name of the file to the server

char fname[50];

strcpy(fname,argv[1]);

int bytesReceived = 0;

char recvBuff[1024];

printf("requested file name: %s\n", fname);

write(sock, fname, 50);

bytesReceived = read(sock, recvBuff, 1024);

// no file matched

if (bytesReceived == 0) {

printf("There is no file in server named: %s\n", fname);

} else if (bytesReceived < 0) {

printf("\n Read Error \n");

} else {

//create a file for receiving

FILE \*fp;

printf("Receiving file...\n");

fp = fopen(fname, "ab");

if(NULL == fp) {

printf("Error opening file");

return 1;

}

// in case the file is less than 1 bytes

printf("writing\n");

fflush(stdout);

fwrite(recvBuff, 1,bytesReceived,fp);

//keep writing

while((bytesReceived = read(sock, recvBuff, 1024)) > 0) {

printf("writing\n");

fflush(stdout);

fwrite(recvBuff, 1,bytesReceived,fp);

}

fclose(fp);

//check the file

printf("File received completely!");

}

} else {

printf("Please enter the file name\n");

}

return 0;

}