#### COEN 146: Computer Networks – Spring 2019

**Lab assignment 3: Programming UDP/IP and TCP/IP socket**

**Objectives**

##### To use the copy one binary file to another

##### To build client/ server applications using UDP/IP socket

##### **Part 1: Copy a binary/ text file to another file**

Problem: Copy a binary/text file to another file.

Analysis:

* Input: pass file names as arguments to main, so your main function would need to defined as follows:

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

Your files: src.dat and dest.dat files.

* File reading can be accomplished by using either:
  + Functions: fopen, fwrite, and fread for binary files or fprintf and fscanf for text files
    - FILE \*fopen(const char \*filename, const char \*mode)
    - fwrite( ptr, int size, int n, FILE \*fp ); or fprintf() (for text files)
    - fread( ptr, int size, int n, FILE \*fp ); or fscanf() (for text files)
    - fclose(ptr);

e.g.

FILE \*fp;

fp = fopen(“src.dat”,"r”);

fp = fopen(“dest.dat","w”);

fwrite(&buf,sizeof(buf),1,fp);

fread(&buf,sizeof(buf),1,fp);

fclose(fp);

OR

* + System calls: open, read, write
    - int open (const char\* Path, int flags [, int mode ]);
    - size\_t read (int fd, void\* buf, size\_t cnt);
    - size\_t write (int fd, void\* buf, size\_t cnt);

e.g.:

int fd = open("foo.txt", O\_RDWR);

           int nw = write(fd, buf, strlen(buf));

int nr = read(fd, buf, 40);

close (fd);

Need to include the following libraries:

* + - #include<sys/types.h>
    - #include<sys/stat.h>
    - #include <fcntl.h>

1. Write your C program to copy files (binary and text) using functions, compile, debug, run, and test
2. Write your C program to copy files (binary and text) using system calls, compile, debug, run, and test

Demonstrate your programs to the TA

##### **Part 2: Client – Server with UDP/IP and TCP/IP**

The following program gives the source code of a UDP client

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <netdb.h>

#include <stdio.h>

#include <unistd.h>

#include <errno.h>

#include <string.h>

#include <stdlib.h>

/\* UDP Client \*/

int main()

{

int sockfd;

struct sockaddr\_in server\_addr;

struct hostent \*host;

char send\_data[1024];

host= (struct hostent \*) gethostbyname((char \*)"127.0.0.1");

if ((sockfd = socket(AF\_INET, SOCK\_DGRAM, 0)) == -1)

{

perror("socket");

exit(1);

}

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

server\_addr.sin\_port = htons(5000);

server\_addr.sin\_addr = \*((struct in\_addr \*)host->h\_addr);

bzero(&(server\_addr.sin\_zero),8);

while (1)

{

printf("Type Something (q or Q to quit):");

gets(send\_data);

if ((strcmp(send\_data , "q") == 0) || strcmp(send\_data , "Q") == 0)

break;

else

sendto(sockfd, send\_data, strlen(send\_data), 0,

(struct sockaddr \*)&server\_addr, sizeof(struct sockaddr));

}

}

1. Write a UDP server that would respond to continuous client requests. You may use the following code snippet if needed. Run, compile and test the client/ server communication and write down your observation. Your file code for both client and server need to be commented in details, explaining the purpose of each line(s) of code. Note: the IP address 127.0.0.1 is a loop back IP address for a local host. i.e. both of your client and server run on the same machine. Demonstrate your program to the TA.

Note: You need to run your client and server programs at the same time, and so run the server first as a background process.

e.g.: ./servUDP&, then type ./clnUDP, where servUDP and clnUDP are your generated executables.

struct sockaddr\_in server\_addr , client\_addr;

if ((sockfd = socket(AF\_INET, SOCK\_DGRAM, 0)) == -1)

{

perror("Socket");

exit(1);

}

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

server\_addr.sin\_port = htons(5000);

server\_addr.sin\_addr.s\_addr = INADDR\_ANY;

bzero(&(server\_addr.sin\_zero),8);

if (bind(sockfd,(struct sockaddr \*)&server\_addr, sizeof(struct sockaddr)) == -1)

{

perror("Bind");

exit(1);

}

addr\_len = sizeof(struct sockaddr);

        printf("UDPServer Waiting for client on port 5000");

fflush(stdout);

     while (1)

     {

bytes\_read = recvfrom(sockfd,recv\_data,1024,0,

     (struct sockaddr \*)&client\_addr, &addr\_len);

     recv\_data[bytes\_read] = '\0';

printf("\n(%s , %d) said : ",inet\_ntoa(client\_addr.sin\_addr),

ntohs(client\_addr.sin\_port));

printf("%s", recv\_data);

     fflush(stdout);

}

1. Now modify your UDP client server program so that your client and server run on two separate machines. In this case, you have to coordinate with one of your classmates to do so.
2. Assign your UDP client to read from src.dat file, sends to UDP server, and then UDP server copies to dest.dat file
3. Modify the UDP client in Step 5 to become a TCP client and the UDP server in Step 5 to become a TCP server.

Demonstrate to the TA and upload your source code to Camino.

**Requirements to complete the lab**

1. Show the TA correct execution of the program you wrote for Part 1 and upload source code to Camino.
2. Show the TA correct execution of the program you wrote for Part 2 (UDP) and upload source code to Camino.
3. Show the TA correct execution of the program you wrote for Part 2 (TCP) and upload source code to Camino.

Be sure to retain copies (machine and/or printed) of your source code. You will want these for study purposes and to resolve any grading questions (should they arise)

Please start each program with a descriptive block that includes minimally the following information:

/\*

\* Name: <your name>

\* Date:

\* Title: Lab3 - Part ….

\* Description: This program … <you should

\* complete an appropriate description here.>

\*/