Computer Network

Assignment : 3

List of Topics: Socket Programming, Inter-Process Communication, fork()

Group No. : 22

Urja Gandhi	202112039
Sharvi Gabani	202112066
Sagar Variya	202112114

Group Chat Application:

1. All clients request with the server.

2. Whenever Server receives any connect request, it makes a child process, and sends child process's handle to the requested client.

3. And when child received any message from the client then it will broadcast that message to all the child process(Hint: Send message to the server then server will broadcast to all it's child processes)

Server1.c :

```c
#include <unistd.h>
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>
#define PORT 8080 //main server port
#define PORT1 8081 //Group 1
#define PORT2 8082 //Group 2

int main(int argc, char const *argv[])
{
	int grp_id1, grp_id2;
	int fd[2][2]; // 2 pipe for 2 group

	int sock_descriptor,clientSocket,recv_msg_size,sendStatus;
	int grp1_port,grp2_port,grp_port;
	int option =1;

	struct sockaddr_in sock_address;
```

```
	char server_msg;
	char *grpNameList="\n Available Groups List : \n GROUP1 \n GROUP2
\n Enter your choice : ";
	char *grpNameInput;
	int socket_addr_len = sizeof(sock_address);

	char *grp1_chat = "";
	char *grp2_chat = "";

	grp1_chat = malloc(5000);
	grp2_chat = malloc(5000);

	char recvClientMsg[1024]={0};
	char grp1_buffer[1024]={0};


	if(pipe(fd[0]) < 0)
	{
		perror("\n (X) Error: Pipe 1 failed");
		exit(EXIT_FAILURE);
	}
	if(pipe(fd[1]) < 0)
	{
		perror("\n (X) Error: Pipe 2 failed");
		exit(EXIT_FAILURE);
	}

	grp_id1 = fork();
	if(grp_id1 > 0)
	{
		grp_id2 = fork();
		if(grp_id2 > 0)
		{
			//Main Server or Parent Process
			if ((sock_descriptor = socket(AF_INET, SOCK_STREAM, 0))
== 0)
			{
				perror("\n (X) Socket failed...");
				exit(EXIT_FAILURE);
			}

			if (setsockopt(sock_descriptor, SOL_SOCKET,
SO_REUSEADDR | SO_REUSEPORT,&option, sizeof(option)))
			{
				perror("\n (X) Setsockopt error...");
				exit(EXIT_FAILURE);
			}

			sock_address.sin_family = AF_INET;
```

```
                  sock_address.sin_addr.s_addr = INADDR_ANY;
                  sock_address.sin_port = htons( PORT );

                  if (bind(sock_descriptor, (struct sockaddr
*)&sock_address,socket_addr_len)<0)
                  {
                        perror("\n (X) Bind failed at Server...");
                        exit(EXIT_FAILURE);
                  }

                  if (listen(sock_descriptor, 3) < 0)
                  {
                        perror("\n (X) Listening error at Server...");
                        exit(EXIT_FAILURE);
                  }

                  printf("\n (#) Listening from Main Server...");

                  while(1){
                        //Pipe for GROUP 1
                        close(fd[0][1]);
                        read(fd[0][0],&grp1_port, sizeof(grp1_port));
                        close(fd[0][0]);

                        //Pipe for GROUP 2
                        close(fd[1][1]);
                        read(fd[1][0],&grp2_port, sizeof(grp2_port));
                        close(fd[1][0]);

                        printf("\n Port receive from Group 1 : %d",
grp1_port);
                        printf("\n Port receive from Group 2 : %d",
grp2_port);

                        if ((clientSocket = accept(sock_descriptor,
(struct sockaddr *)&sock_address,&socket_addr_len))<0)
                        {
                              perror("\n (X) Accept Error at Main
Server");
                              exit(EXIT_FAILURE);
                        }

                        printf("\n (#) Accepted in Main Server...\n");
                        recv_msg_size=read( clientSocket , recvClientMsg,
1024);
                        printf("\n Client : %s\n",recvClientMsg);
                        memset(recvClientMsg,0,1024);

                        //sending group list to client
                        sendStatus=send(clientSocket, grpNameList,
```

```
strlen(grpNameList), 0);

                        //reading which group to join
                        recv_msg_size=read( clientSocket , recvClientMsg,
1024);
                        //compare the group string
                        if(strcmp(recvClientMsg,"JOIN GROUP 1")==0){
                              grp_port=grp1_port;
                              printf("\n Client : %s\n",recvClientMsg);
                        }else if(strcmp(recvClientMsg,"JOIN GROUP
2")==0){
                              grp_port=grp2_port;
                              printf("\n Client : %s\n",recvClientMsg);
                        }
                        else{
                              printf("\n (X) You have entered wrong
group...\n");
                        }

                        //send the port of child to client
                        if(write(clientSocket, &grp_port,
sizeof(grp_port))<0){
                              perror("\n (X) Error in sending to
client");
                        }
                        printf("\n Port sent to client : %d",grp_port);

                  }

            }
            else if(grp_id2 == 0){
                  //child 2 / Group 2

                  printf("\n (#) Inside Group2 chat room (Child process
2) \n");
                  int c2_sock_desc, c2_client_sock, opt=1;
                  struct sockaddr_in c2_address;
                  int c2_addrlen = sizeof(c2_address);
                  char *msg;
                  char grp2_buffer[1024]={0};

                  if((c2_sock_desc = socket(AF_INET, SOCK_STREAM, 0)) <=
0)
                  {
                        perror("\n (X) Socket creation failed at group 2
server");
                        exit(EXIT_FAILURE);
                  }

                  printf("\n (#) Socket creation successful in Group 2
```

```
server...\n");

                  if (setsockopt(c2_sock_desc, SOL_SOCKET, SO_REUSEADDR |
SO_REUSEPORT, &opt, sizeof(opt)))
                  {
                        perror("\n (X) setsockopt error at group 2
server");
                        exit(EXIT_FAILURE);
                  }

                  c2_address.sin_family = AF_INET;
                  c2_address.sin_addr.s_addr = INADDR_ANY;
                  c2_address.sin_port = htons(PORT2);

                  if(bind(c2_sock_desc, (struct sockaddr *)&c2_address,
c2_addrlen) < 0)
                  {
                        perror("\n (X) bind failed at group 2 server");
                        exit(EXIT_FAILURE);
                  }

                  if(listen(c2_sock_desc, 5) < 0)
                  {
                        perror("\n (X) listening failed at group 2
server");
                        exit(EXIT_FAILURE);
                  }

                  while(1){
                        int group2Port = PORT2;
                        close(fd[1][0]);
                        write(fd[1][1], &group2Port, sizeof(group2Port));
                        close(fd[1][1]);

                        if((c2_client_sock =  accept(c2_sock_desc,
(struct sockaddr * )&c2_address, (socklen_t*)&c2_addrlen) )< 0)
                        {
                              perror("\n (X) can't accept at group 2
server");
                              exit(EXIT_FAILURE);
                        }

                        printf("\n (#) Connection accepted from client in
Group chat 2 \n");

                        if(strlen(grp2_chat) == 0)
                        {
                              printf("\n (X) No messages in history were
found \n");
                              send(c2_client_sock, "---- Welcome to
```

```
Group2 ----", 28, 0);
                        }
                        else
                        {
                              send(c2_client_sock, grp2_chat,
strlen(grp2_chat), 0);
                        }

                        while(1)
                        {
                              read( c2_client_sock , grp2_buffer, 1024);
                              if(!strcmp(grp2_buffer,"exit"))
                              {
                                    printf(" (#) Client disconnected with
Group 2 Server...\n");
                                    break;
                              }

                              printf("\nClient : %s\n",grp2_buffer );//if
any message send from server then it will print

                              strcat(grp2_chat,grp2_buffer);
                              memset(grp2_buffer, 0, 1024);
                        }
                  }
            }
      }
      else
      {
            printf("\n Inside Group1 chat room (Child process 1)\n");
                  //child 1 / Group 1

                  printf("\n (#) Inside Group1 chat room (Child process
1) \n");
                  int c1_sock_desc, c1_client_sock, opt=1;
                  struct sockaddr_in c1_address;
                  int c1_addrlen = sizeof(c1_address);
                  char *msg;
                  char grp1_buffer[1024]={0};

                  if((c1_sock_desc = socket(AF_INET, SOCK_STREAM, 0)) <=
0)
                  {
                        perror("\n (X) Socket creation failed at group 1
server");
                        exit(EXIT_FAILURE);
                  }

                  printf("\n (#) Socket creation successful in Group 1
server...\n");
```

```
                if (setsockopt(c1_sock_desc, SOL_SOCKET, SO_REUSEADDR |
SO_REUSEPORT, &opt, sizeof(opt)))
                {
                      perror("\n (X) setsockopt error at group 1
server");
                      exit(EXIT_FAILURE);
                }

                c1_address.sin_family = AF_INET;
                c1_address.sin_addr.s_addr = INADDR_ANY;
                c1_address.sin_port = htons(PORT1); // 8081

                if(bind(c1_sock_desc, (struct sockaddr *)&c1_address,
c1_addrlen) < 0)
                {
                      perror("\n (X) bind failed at group 2 server");
                      exit(EXIT_FAILURE);
                }

                if(listen(c1_sock_desc, 5) < 0)
                {
                      perror("\n (X) listening failed at group 2
server");
                      exit(EXIT_FAILURE);
                }

                while(1){
                      int group1Port = PORT1;
                      close(fd[0][0]);
                      write(fd[0][1], &group1Port, sizeof(group1Port));
                      close(fd[0][1]);

                      if((c1_client_sock =  accept(c1_sock_desc,
(struct sockaddr * )&c1_address, (socklen_t*)&c1_addrlen) )< 0)
                      {
                            perror("\n (X) can't accept at group 1
server");
                            exit(EXIT_FAILURE);
                      }

                      printf("\n (#) Connection accepted from client in
Group chat 1 \n");

                      if(strlen(grp1_chat) == 0)
                      {
                            printf("\n (X) No messages in history were
found \n");
                            send(c1_client_sock, "---- Welcome to
Group1 ----", 28, 0);
```

```
                        }
                        else
                        {
                                send(c1_client_sock, grp1_chat,
strlen(grp1_chat), 0);
                        }

                        while(1)
                        {
                                read( c1_client_sock , grp1_buffer, 1024);
                                if(!strcmp(grp1_buffer,"exit"))
                                {
                                        printf(" (#) Client disconnected with
Group 1 Server...\n");
                                        break;
                                }

                                printf("\nClient : %s\n",grp1_buffer );//if
any message send from server then it will print

                                strcat(grp1_chat,grp1_buffer);
                                strcat(grp1_chat, "\n");
                                memset(grp1_buffer, 0, 1024);
                        }
                }
        }
        return 0;
}
```

Client2.c :

```
// Client side C/C++ program to demonstrate Socket programming

#include <stdio.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <string.h>
#define PORT 8080

int main(int argc, char const *argv[])
{
	int sock_descriptor = 0, recv_msg_size;
	struct sockaddr_in server_addr;
	int socket_addr_len = sizeof(server_addr);
	char buffer[1024] = {0};
	char *client_msg;
	int option;
	int grp_port; // group port to join

	if ((sock_descriptor = socket(AF_INET, SOCK_STREAM, 0)) < 0)
	{
		printf("\n (X) Socket creation error... \n");
		return -1;
	}

	server_addr.sin_family = AF_INET;
	server_addr.sin_port = htons(PORT);

	if(inet_pton(AF_INET, "127.0.0.1", &server_addr.sin_addr)<=0)
	{
		printf("\n (X) Invalid address/ Address not supported...
\n");
		return -1;
	}

	if (connect(sock_descriptor, (struct sockaddr *)&server_addr,
socket_addr_len) < 0)
	{
		printf("\n (X) Connection Failed... \n");
		return -1;
	}

	printf("\n (#) Client connected to main server... \n");

	gets(client_msg); //input your name
	send(sock_descriptor , client_msg , strlen(client_msg) , 0 ); //
```

```
send name given by client to server
      //receive group list
      recv_msg_size = read(sock_descriptor , buffer, 1024);

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

      memset(buffer, 0, 1024);

      gets(client_msg);
      send(sock_descriptor , client_msg , strlen(client_msg) , 0 ); //
send msg for joining group

      recv_msg_size = read(sock_descriptor , &grp_port,
sizeof(grp_port));
      printf("\n (#) PORT receive : %d",grp_port);

      close(sock_descriptor); //disconnect from main server

      printf("\n (#) Client disconnected from Main Server...");

      if ((sock_descriptor = socket(AF_INET, SOCK_STREAM, 0)) < 0)
      {
            printf("\n (X) Socket creation error... \n");
            return -1;
      }

      server_addr.sin_family = AF_INET;
      server_addr.sin_port = htons(grp_port); // Group port receive from
chosen child group

      if(inet_pton(AF_INET, "127.0.0.1", &server_addr.sin_addr)<=0)
      {
            printf("\n (X) Invalid address/ Address not supported...
\n");
            return -1;
      }

      if (connect(sock_descriptor, (struct sockaddr *)&server_addr,
socket_addr_len) < 0)
      {
            printf("\n (X) Connection Failed... \n");
            return -1;
      }

      printf("\n (#) Client connected to Group Server...");

      recv_msg_size = read(sock_descriptor , buffer, 1024);

      printf("\n Server : %s\n",buffer);
```

```
        memset(buffer,0,2024);
        while(1){
                gets(client_msg); // taking input for msg
                if(!strcmp(client_msg,"exit")){
                        send(sock_descriptor,client_msg,strlen(client_msg),0);
                        printf("\n (X) Client disconnected from Group Server");
                        close(sock_descriptor); // disconnect from group
                        break;
                }

                send(sock_descriptor,client_msg,strlen(client_msg),0);
        }
        return 0;
}
```

Output :

Serverside Terminal Output :

```
ubuntu22@ubuntu22-VirtualBox:~/CN/Lab3/Apna_final$ gcc server1.c -o server
ubuntu22@ubuntu22-VirtualBox:~/CN/Lab3/Apna_final$ ./server

 Inside Group1 chat room (Child process 1)

 (#) Inside Group1 chat room (Child process 1)

 (#) Socket creation successful in Group 1 server...

 (#) Inside Group2 chat room (Child process 2)


 (#) Socket creation successful in Group 2 server...
 (#) Listening from Main Server...
 Port recieve from Group 1 : 8081
 Port recieve from Group 2 : 8082
 (#) Accepted in Main Server...

 Client : Urja here

 Client : JOIN GROUP 2

 Port sent to client : 8082
 Port recieve from Group 1 : 8081

 (#) Connection accepted from client in Group chat 2

 (X) No messages in history were found

Client : hello world

Client : Good morning
 (#) Client disconnected with Group 2 Server...
```

```
 Port recieve from Group 2 : 8082
 (#) Accepted in Main Server...

 Client : Sharvi here2

 Client : JOIN GROUP 1


 (#) Connection accepted from client in Group chat 1

 (X) No messages in history were found
 Port sent to client : 8081
 Port recieve from Group 1 : 8081

Client : hello earth

Client : Good night
 (#) Client disconnected with Group 1 Server...
 Port recieve from Group 2 : 8082
 (#) Accepted in Main Server...

 Client : Sagar here 1

 Client : JOIN GROUP 2

 Port sent to client : 8082

 Port recieve from Group 1 : 8081
 (#) Connection accepted from client in Group chat 2

Client : Good afternoon
 (#) Client disconnected with Group 2 Server...
```

```
 Port recieve from Group 2 : 8082
 (#) Accepted in Main Server...

 Client : Khushi here2

 Client : JOIN GROUP 2


 (#) Connection accepted from client in Group chat 2
 Port sent to client : 8082
 Port recieve from Group 1 : 8081
```

Clientside terminal Output :

```
ubuntu22@ubuntu22-VirtualBox:~/CN/Lab3/Apna_final$ ./client

 (#) Client connected to main server...
Urja here

 Server :
 Available Groups List :
 GROUP1
 GROUP2
 Enter your choice : JOIN GROUP 2

 (#) PORT recieve : 8082
 (#) Client disconnected from Main Server...
 (#) Client connected to Group Server...
 Server : ---- Welcome to Group2 ----
hello world
Good morning
exit
```

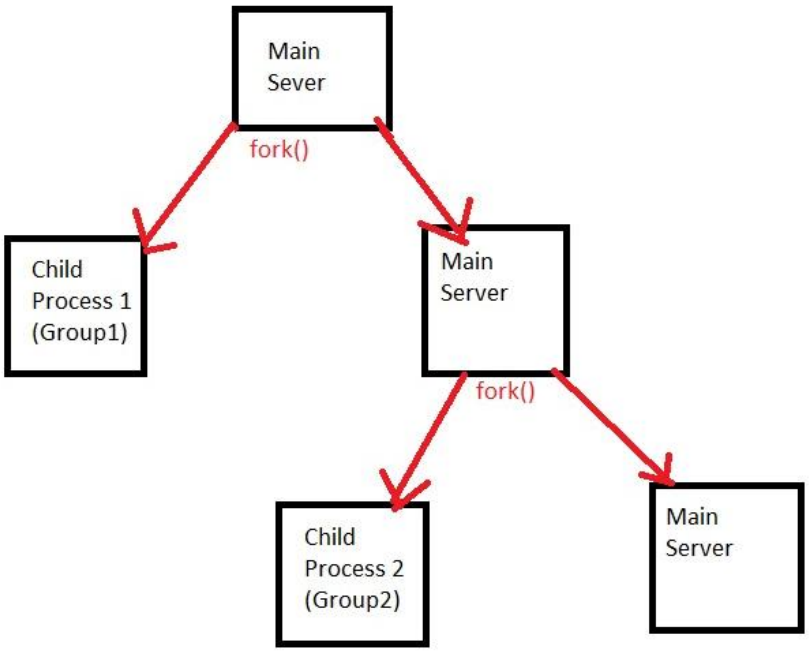
```
ubuntu22@ubuntu22-VirtualBox:~/CN/Lab3/Apna_final$ ./client

 (#) Client connected to main server...
Sharvi here

 Server :
 Available Groups List :
 GROUP1
 GROUP2
 Enter your choice : JOIN GROUP 1

 (#) PORT recieve : 8081
 (#) Client disconnected from Main Server...
 (#) Client connected to Group Server...
 Server : ---- Welcome to Group1 ----
hello earth
Good night
exit
```

```
ubuntu22@ubuntu22-VirtualBox:~/CN/Lab3/Apna_final$ ./client

 (#) Client connected to main server...
Sagar here

 Server :
 Available Groups List :
 GROUP1
 GROUP2
 Enter your choice : JOIN GROUP 2

 (#) PORT recieve : 8082
 (#) Client disconnected from Main Server...
 (#) Client connected to Group Server...
 Server : hello worldGood morning
Good afternoon
exit
```

```
ubuntu22@ubuntu22-VirtualBox:~/CN/Lab3/Apna_final$ ./client

 (#) Client connected to main server...
Khushi here

 Server :
 Available Groups List :
 GROUP1
 GROUP2
 Enter your choice : JOIN GROUP 2

 (#) PORT recieve : 8082
 (#) Client disconnected from Main Server...
 (#) Client connected to Group Server...
 Server : hello worldGood morningGood afternoon
```

Explanation :

- As per the requirement, this group chat application will accept multiple client connect request and give them feature to join a particular Group. We have made 2 child processes which represent two groups using fork() system call as shown as the figure below.



- In the beginning, client will connect to the main server which uses PORT 8080.
- Then client will choose the Group it wants to join.
- The main server (parent process) will read PORT number which are used by the child process 1 and child process 2 using pipe() system call.
- Now, parent process will send the required PORT number which is chosen by the client as Group No. in which it wants to communicate.
- Client will now disconnect with Main Server and form connection with Child process (Group) whose PORT number it has receive.
- Client will receive the previous messages if any. And it will send its own messages.
- Client exits after sending messages but Main Sever will be in listening mode to accept other client request.