

Lab Assignment-2

CSN-361: Computer Networks Laboratory

THEEGALA LOKESH REDDY

Enrollment No:-17114073

B.tech CSE 3 rd Year.

Problem Statement 1:

Write a socket program in C to connect two nodes on a network to communicate with each other, where one socket listens on a particular port at an IP, while other socket reaches out to the other to form a connection.

-

Data Structure and Functions used:-

For Server -

1. Socket creation:

struct sockaddr_in servaddr : sockaddr_in structure for storing server address

```
sockfd = socket(AF_INET, SOCK_STREAM, 0);
```

sockfd: socket descriptor, an integer (like a file-handle)

AF_INET (eg IPv4 protocol) is domain

SOCK_STREAM: TCP(reliable, connection oriented)

SOCK_DGRAM: UDP(unreliable, connectionless)

protocol: Protocol value for Internet Protocol(IP), which is 0 , confirms it is working

2. Bind:

```
(bind(sockfd, (SA*)&servaddr, sizeof(servaddr)))
```

After creation of the socket, bind function binds the socket to the address and port number

specified in addr. In this we bind the server to the localhost, hence we use INADDR_ANY to specify the IP address of our system .

3. Listen:

```
(listen(sockfd, n))_
```

It puts the server socket in waiting for the client to approach the server to make a connection. Here n defines the no of clients u can connect.

4. Accept:

```
connfd = accept(sockfd, (SA*)&cli, &len);
```

It extracts the first connection request for the listening socket, sockfd, creates a new connected socket, and returns a new file descriptor referring to that socket. At this point, connection is established between client and server, and they are ready to transfer data.

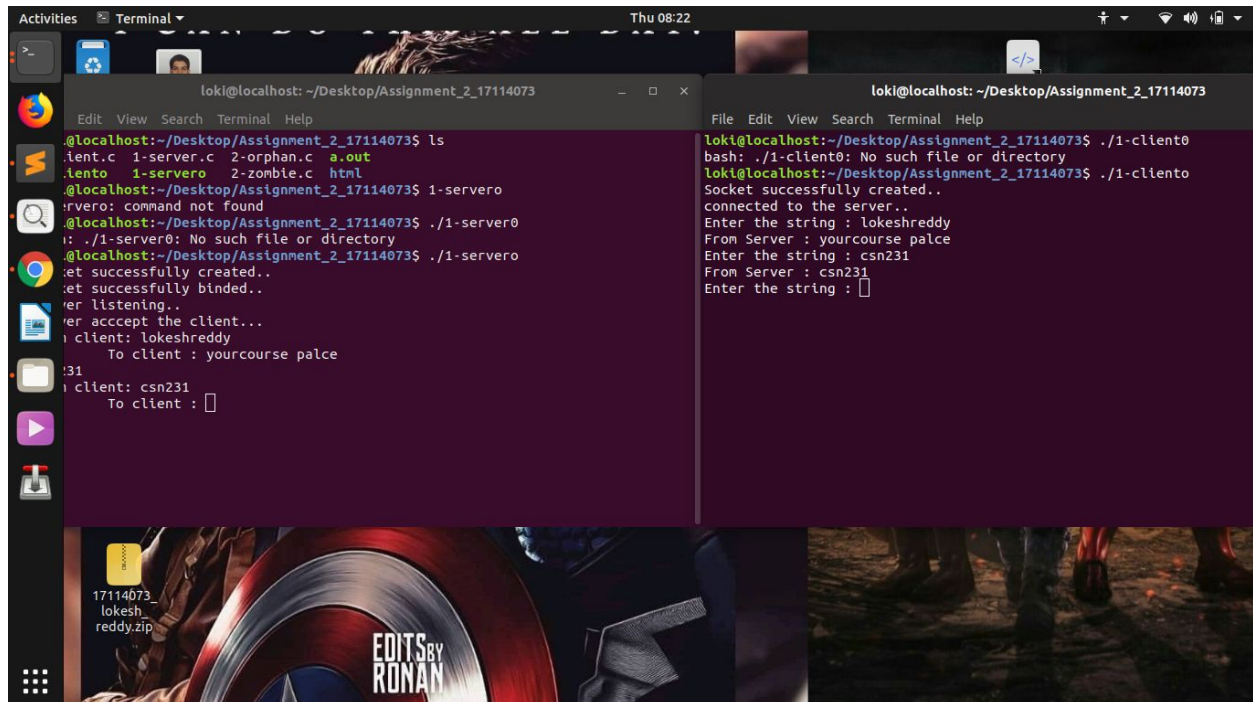
For Client:-

1. Socket connection: Exactly same as that of server's socket creation

2. Connect:

```
connect(sockfd, (SA*)&servaddr, sizeof(servaddr);
```

The connect() system call connects the socket referred to by the file descriptor sockfd to the address specified by addr. Server's address and port is specified in Addr



```
loki@localhost: ~/Desktop/Assignment_2_17114073
loki@localhost:~/Desktop/Assignment_2_17114073$ ls
client.c 1-server.c 2-orphan.c a.out
lento 1-servero 2-zombie.c html
loki@localhost:~/Desktop/Assignment_2_17114073$ 1-servero
servero: command not found
loki@localhost:~/Desktop/Assignment_2_17114073$ ./1-servero
./1-servero: No such file or directory
loki@localhost:~/Desktop/Assignment_2_17114073$ ./1-servero
set successfully created..
set successfully binded..
server listening..
server accept the client...
client: lokeshreddy
To client : yourcourse palce
csn231
client: csn231
To client :

loki@localhost:~/Desktop/Assignment_2_17114073$ ./1-cliento
bash: ./1-cliento: No such file or directory
loki@localhost:~/Desktop/Assignment_2_17114073$ ./1-cliento
Socket successfully created..
connected to the server..
Enter the string : lokeshreddy
From Server : yourcourse palce
Enter the string : csn231
From Server : csn231
Enter the string :
```

Problem Statement 2:

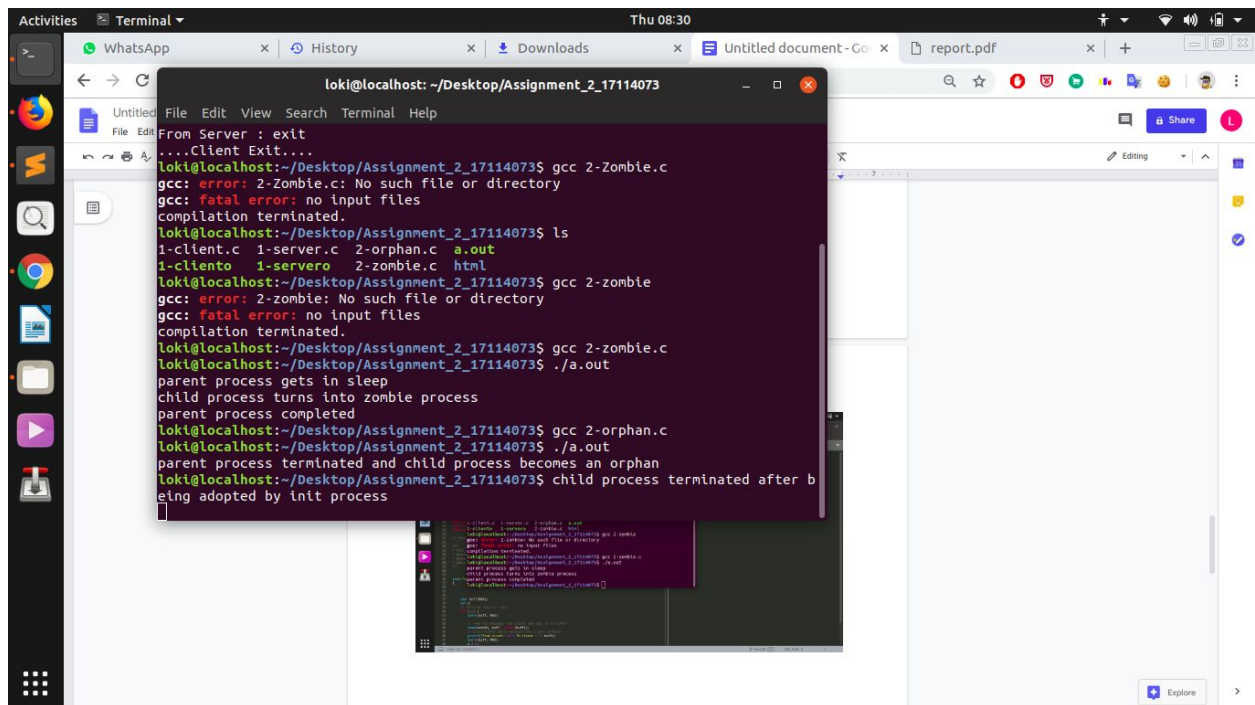
Write a C program to demonstrate both Zombie and Orphan process

Zombie:

Data Structure and Functions used:-

- 1.fork() is used for creating new child of parent.
2. exit() parent finishes its execution using it .
- 3.int Child_pid is used for storing child id.
4. sleep() used for child process to wait

Brief: when parent completes its process and exits without taking exit status from child, child doesn't have parent after that init will adopt the child.



The screenshot shows a Linux desktop environment. In the foreground, a terminal window titled 'loki@localhost: ~/Desktop/Assignment_2_17114073' displays the following commands and output:

```
From Server : exit
....Client Exit....
loki@localhost:~/Desktop/Assignment_2_17114073$ gcc 2-Zombie.c
gcc: error: 2-Zombie.c: No such file or directory
gcc: fatal error: no input files
compilation terminated.
loki@localhost:~/Desktop/Assignment_2_17114073$ ls
1-client.c 1-server.c 2-orphan.c a.out
1-cliento 1-servero 2-zombie.c html
loki@localhost:~/Desktop/Assignment_2_17114073$ gcc 2-zombie
gcc: error: 2-zombie: No such file or directory
gcc: fatal error: no input files
compilation terminated.
loki@localhost:~/Desktop/Assignment_2_17114073$ gcc 2-zombie.c
loki@localhost:~/Desktop/Assignment_2_17114073$ ./a.out
parent process gets in sleep
child process turns into zombie process
parent process completed
loki@localhost:~/Desktop/Assignment_2_17114073$ gcc 2-orphan.c
loki@localhost:~/Desktop/Assignment_2_17114073$ ./a.out
parent process terminated and child process becomes an orphan
loki@localhost:~/Desktop/Assignment_2_17114073$ child process terminated after b
eing adopted by init process
```

In the background, a web browser window is open, showing a document titled 'report.pdf'. The desktop also features a sidebar with application icons and a top bar with system status indicators.

screen shots of doxygen:

Activities Google Chrome Thu 08:35

WhatsApp x History x Downloads x Untitled docu x report.pdf x Web series x My Project: Fil x

File | /home/loki/Desktop/Assignment_2_17114073/html/files.html

My Project 1

Assignment_2_17114073

Main Page Files Search

My Project

- Files
 - File List
 - File Members

File List

Here is a list of all files with brief descriptions:

1-client.c	
1-server.c	
2-orphan.c	
2-zombie.c	

Generated by **doxygen** 1.8.13

Activities Google Chrome Thu 08:35

WhatsApp x History x Downloads x Untitled docu x report.pdf x Web series x My Project: Fil x

File | /home/loki/Desktop/Assignment_2_17114073/html/globals.html

My Project 1

Assignment_2_17114073

Main Page Files Search

My Project

- Files
 - File List
 - File Members

All

Functions

Macros

File Members

Here is a list of all file members with links to the files they belong to:

- func() : 1-client.c , 1-server.c
- main() : 1-client.c , 1-server.c , 2-orphan.c , 2-zombie.c
- MAX : 1-client.c , 1-server.c
- PORT : 1-client.c , 1-server.c
- SA : 1-client.c , 1-server.c

Generated by **doxygen** 1.8.13

