Mayank Lakhotia
2022PMN4226

Practical – 4

Aim: -

Implement TWO-WAY Inter-process communication using FIFOs. Consider TWO independent processes for communication. This communication must continue till a specific key is pressed or a STOP message is sent by any one of the processes.

Theory: -

FIFO: A named pipe (also known as a **FIFO**) is one of the methods for inter-process communication. It is an extension of tradition pipe on UNIX. A traditional pipe is unnamed and lasts as long as the process, whereas named pipe can last as long as the system is up and it can be deleted if no longer in use.

Named pipe appears as a file. A FIFO file is a special file on local storage which allows two or more processes to communicate with each other by reading/writing to /from this file.

As named pipe (FIFO) is a kind of file, we can use all the system calls associated with it i.e. **open, read, write, close.**

A **FIFO** special file is entered into the file system by calling **mkfifo()** in C. Once we have created a FIFO special file in this way, any process can open it for reading or writing, in the same way as an ordinary file. However, it has to be open at both ends simultaneously before you can proceed to do any input or output operations on it.

Syntax of mkfifo()
mkfifo(const char *pathname, mode_t mode);

where mode describes the permission. It is modified by the process's umask in the usual way: the permissions of the created file are (mode & ~umask).

A named pipe (also known as a FIFO) is one of the methods for intern-process communication.
• It is an extension to the traditional pipe concept on Unix. A traditional pipe is "unnamed" and lasts only as long as the process.
• A named pipe, however, can last as long as the system is up, beyond the life of the process. It can be deleted if no longer used.
• Usually, a named pipe appears as a file and generally processes attach to it for inter-process communication. A FIFO file is a special kind of file on the local storage which allows two or more processes to communicate with each other by reading/writing to/from this file.
• A FIFO special file is entered into the filesystem by calling *mkfifo ()* in C or Python. Once we have created a FIFO special file in this way, any process can open it for reading or writing, in the same way as an ordinary file. However, it has to be open at

both ends simultaneously before you can proceed to do any input or output operations on it.

Creating a FIFO file: In order to create a FIFO file, a function calls i.e., mkfifo or mknod is used.

Source Code: - C Language

client.c

```
#include<stdio.h>

#include<signal.h>

#include<sys/types.h>

#include<unistd.h>

#include<stdlib.h>

#include<string.h>

#include<fcntl.h>

#include<sys/stat.h>

#include<sys/shm.h>

#include<sys/ipc.h>

#include<sys/sem.h>


int main() {


    int file1,file2;

    char a[20];


    mkfifo("f1",0666);

    perror("Client");

    mkfifo("f2",0666);

    perror("Server");


    file1 = open("f1", O_RDWR);

    file2= open("f2", O_RDWR);
    //printf("ID = %d\n",getpid());

    if(fork())
    {
        while(1)
        {
            printf("Enter a message to server...\n");
            scanf("%s",a);
            write(file1,a,strlen(a)+1);
        }
    }

    else
    {
        while(1)
        {
            read(file2,a,sizeof(a));
            printf("Message received from server = %s\n",a);
        }
    }
}
```

server.c

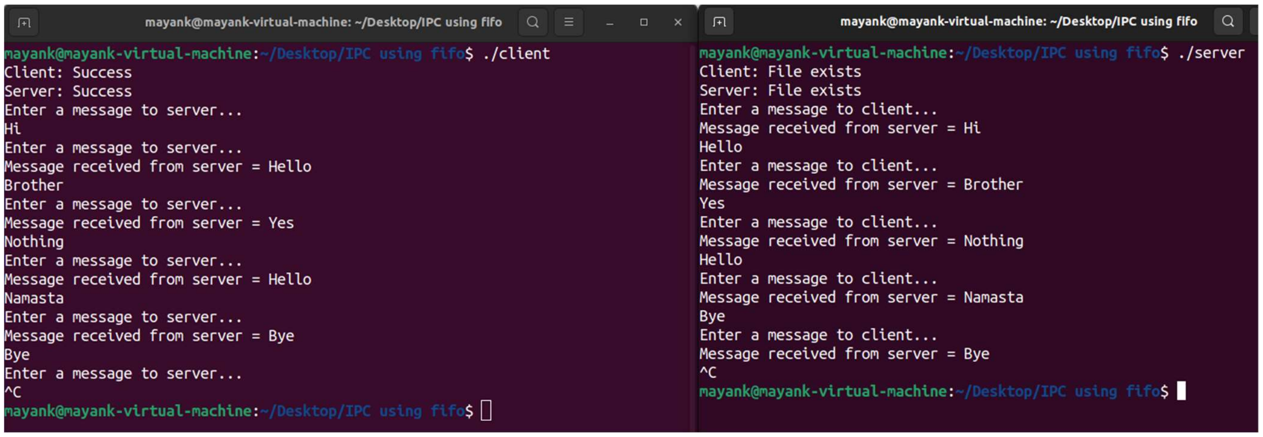
```
#include<stdio.h>
#include<signal.h>
#include<sys/types.h>
#include<unistd.h>
#include<stdlib.h>
#include<string.h>
#include<fcntl.h>
```

Mayank Lakhotia
2022PMN4226

```
#include<sys/stat.h>
#include<sys/shm.h>
#include<sys/ipc.h>
#include<sys/sem.h>

int main() {

    int file1,file2;
    char a[20];

    mkfifo("f1",0666);
    perror("Client");
    mkfifo("f2",0666);
    perror("Server");

    file1= open("f1", O_RDWR);
    file2= open("f2", O_RDWR);

    //printf("ID = %d\n",getpid());

    if(fork())
    {
        while(1)
        {
            read(file1,a,sizeof(a));
            printf("Message received from server = %s\n",a);
        }
    }

    else
    {
        while(1)
        {
            printf("Enter a message to client...\n");
            scanf("%s",a);
            write(file2,a,strlen(a)+1);
        }
    }
}
```

Output: -

Keyboard Interrupt (Stop the chat from both side either from client side or from server side)



```
mayank@mayank-virtual-machine:~/Desktop/IPC using fifo$ ./client
Client: Success
Server: Success
Enter a message to server...
Hi
Enter a message to server...
Message received from server = Hello
Brother
Enter a message to server...
Message received from server = Yes
Nothing
Enter a message to server...
Message received from server = Hello
Namasta
Enter a message to server...
Message received from server = Bye
Bye
Enter a message to server...
^C
mayank@mayank-virtual-machine:~/Desktop/IPC using fifo$
```

```
mayank@mayank-virtual-machine:~/Desktop/IPC using fifo$ ./server
Client: File exists
Server: File exists
Enter a message to client...
Message received from server = Hi
Hello
Enter a message to client...
Message received from server = Brother
Yes
Enter a message to client...
Message received from server = Nothing
Hello
Enter a message to client...
Message received from server = Namasta
Bye
Enter a message to client...
Message received from server = Bye
^C
mayank@mayank-virtual-machine:~/Desktop/IPC using fifo$
```