All about Cdac Hyderabad - DESD

Friday, 21 October 2016

Name Pipes

Now we know about IPC and PIPE.
And we also know that Pipe has two cases for which they are used.
Inter Process communication for Related process and Unrelated process.
Read about these **here (click to read)**

How can we communicate if the processes are unrelated ?
Here Named Pipes comes into picture :
Lets see how -
A named pipe is a special type of file (remember that everything in Linux is a file! EXCEPT 1 place - sockets are not files).

I will try to explain FIFO with a simple example.

Suppose that I need to pass a secret message to my girlfriend(which i don't have) in CDAC. And there is not any mobile or any way to communicate her (Nothing means nothing)
Remember the old days when romeos used to communicate with their Juliets in early 90's. I will use the same way to communicate.

What I will do ?? I will create a secret LOVE_BOX (a simple box) in PC no 12 of CDAC Lab.

I will OPEN the LOVE_BOX and will WRITE my secret message(DATA)(message written a piece of paper) into the LOVE_BOX.

What my girlfriend will do now She will come to cdac,will go to that PC NO 12 and will OPEN that Box and will Read that message.

Here i have options with me while opening the LOVE_BOX and the options are :
I want to open the box for reading or writing ,or want to wait to place until my gf comes to read the box , etc (read man page of fifo for more details of modes).

Now come to the topic. Fifo uses similar concept.

Just replace me with PROCESS 1 and my gf with PROCESS 2.
LOVE_BOX with FIFO and OPEN, READ WRITE TO BE SAME.

Lets see how FIFO works.
Step 1:

Create a fifo using
mkfifo filename (from command line)

Or using mkfifo system call.

```
#include <sys/types.h>
#include <sys/stat.h>
int mkfifo(const char *filename, mode_t mode);
```

ode_t mode = permissions.

Creating A fifo....

Blog Archive

- ▼ 2016 (5)
  - ▼ October (5)
    - OS Moduels Exam Import questions
    - Monolithic kernel VS Micro Kernel
    - OS Last Minute Notes - Part 1
    - Name Pipes
    - All About Inter-Process Communication

```c
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>

int main()
{
    int res = mkfifo("/tmp/my_fifo", 0777);
    if (res == 0) printf("FIFO created\n");
    exit(EXIT_SUCCESS);
}
```

you can see the created fifo also. After running this program go to /tmp/my_fifo folder and type

ls -alF or you just type ls -lF /tmp/my_fifo

Step -2:
Once this fifo is created its is same as a simple .txt file. How we write or read to or from a file? Same way we can use FIFO. Any one proceses will create it and one process will write another will read from the same fifo and can close it at the end same as do with txt files.

Note - Named Pipes are unidirectional. Means that while opening the pipe for communication we need to say that we want to read from it or write from it. We can't read and write to the same fifo at a time. Confused ??

Ok lets go back to the previous example. I have created a LOVE_Box . I went there and OPENED (in write mode) it to write in it. I will be able to do it. Now suppose after some time i come back and want to READ from the box. **This is not allowed.** (As the fifo was opened earlier to write mode).

If i need to communicate in duplex mode i need to either close the fifo and open again in read mode. or Simply use two fifo - open to write and one to read.

The main restriction on opening FIFOs is that a program may not open a FIFO for reading and writing with the mode O_RDWR. If a program violates this restriction, the result is undefined. This is quite a sensi-ble restriction because, normally you use a FIFO only for passing data in a single direction, so there is
no need for an O_RDWR mode.
A process would read its own output back from a pipe if it were opened
read/write.
If you do want to pass data in both directions between programs, it is much better to use either a pair of FIFOs or pipes, one for each direction, or (unusually) explicitly change the direction of the data flow by closing and reopening the FIFO.

At the end of the program we need to first close that created fifo and delete it also using
unlink(fifo_name);

Hope this may help you.
Feel free to ask any doubts in Cdac hyd fb group.

No comments:

Post a Comment

Enter your comment...

Comment as: Mohit Ingale (G ▾ Sign out

Publish Preview Notify me

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)