

Linux Internals & Networking

System programming using Kernel interfaces

Team Emertxe



Contents



Linux Internals & Networking

Contents

- .Introduction
- .Transition to OS programmer
- .System Calls
- .Process
- .IPC
- .Signals
- .Networking
- .Threads
- .Synchronization
- .Process Management
- .Memory Management



Inter Process Communications (IPC)



Inter Process Communications

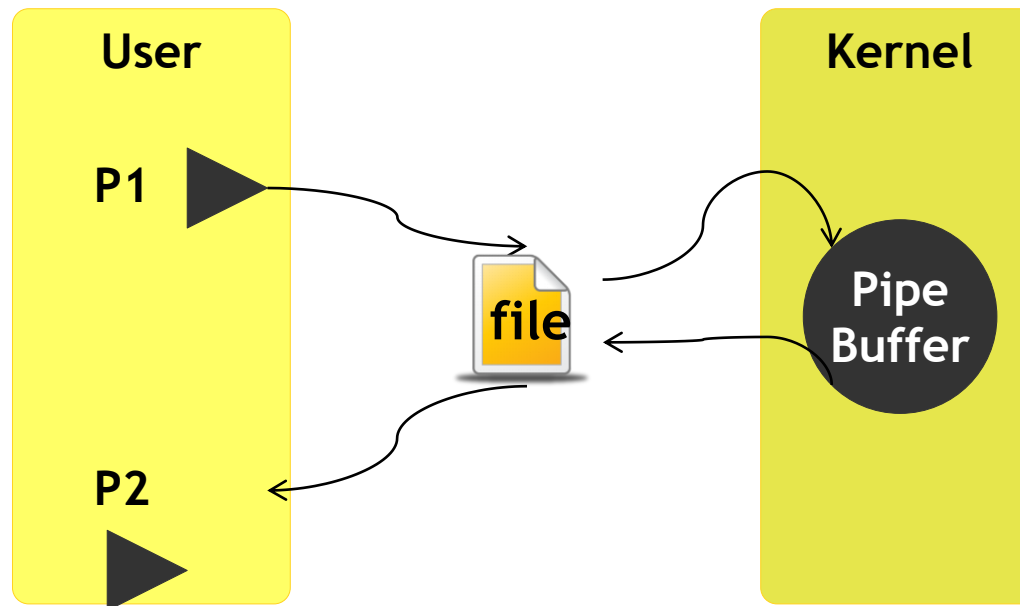
FIFO - Properties



- A *first-in, first-out (FIFO)* file is a pipe that has a name in the file-system
- FIFO file is a pipe that has a name in the file-system
- FIFOs are also called Named Pipes
- FIFOs is designed to let them get around one of the shortcomings of normal pipes

Inter Process Communications

FIFO - Working



Inter Process Communications

FIFO - Creation



- FIFO can also be created similar to directory/file creation with special parameters & permissions
- After creating FIFO, read & write can be performed into it just like any other normal file
- Finally, a device number is passed. This is ignored when creating a FIFO, so you can put anything you want in there
- Subsequently FIFO can be closed like a file

Function	Meaning
<code>int mknod(const char *path, mode_t mode, dev_t dev)</code>	<ul style="list-style-type: none">✓ path: Where the FIFO needs to be created (Ex: "/tmp/Emertxe")✓ mode: Permission, similar to files (Ex: 0666)✓ dev: can be zero for FIFO

Inter Process Communications

FIFO - Access



- Access a FIFO just like an ordinary file
- To communicate through a FIFO, one program must open it for writing, and another program must open it for reading
- Either low-level I/O functions (open, write, read, close and so on) or C library I/O functions (fopen, fprintf, fscanf, fclose, and so on) may be used

```
user@user:~] ls -l myfifo  
prw-rw-r-- 1      satya satya          0 Mar 8  17:36 myfifo
```

prw-

Inter Process Communications

FIFO vs PIPES



- Unlike pipes, FIFOs are not temporary objects, they are entities in the file-system
- Any process can open or close the FIFO
- The processes on either end of the pipe need not be related to each other
- When all I/O is done by sharing processes, the named pipe remains in the file system for later use

Inter Process Communications

FIFO - Example



.Unrelated process can communicate with FIFO

Shell 1

```
user@user:~] cat > /tmp/my_fifo  
Hai hello
```

Shell 2

```
user@user:~] cat /tmp/my_fifo  
Hai hello
```

Inter Process Communications

FIFO - Pros & Cons



PROS

- Naturally synchronized
- Simple to use and create
- Unrelated process can communicate.
- No extra system calls required to communicate (read/write)
- Work like normal file

CONS

- Less memory size (4K)
- Only two process can communicate
- One directional communication
- Kernel is involved

Inter Process Communications

Summary



.We have covered

Data exchange

Communication

.Pipes

.FIFO

.Shared memory

.Signals

.Sockets

Resource usage/access/control

Synchronization

.Semaphores

Team Emertxe



Thank You