Operating System

Linux Boot Process

Filesystems

RAID

IPC

Sockets

RPC

Signals

POSIX signals

Threads

Streams

Caching

Scheduler

Write-through, write-around, write-b...

Tempfiles

System Calls

PID Definition

Shared Memory

Zombie process

What's in an inode?

What are the common run levels in li...

What is the difference between soft ...

What happens when a process forks?

How-To Compile Linux Kernel from ...

Understanding Sticky-Bit, SUID and ...

Understanding User and Kernel Mode

Understanding Linux CPU Load - wh...

The child gets a copy of the stack & program, and there are two processes running. You differentiate the child from the parent by fork's ending status. If zero, you're in the child. If not, you're in the parent and the ending status is the PID of the child.

What happens when a process forks?

- A process calling fork()spawns a child process.
- The child is almost an identical clone of the parent:
- Program Text (segment .text)
- · Stack (ss)

- PCB (eg. registers)
- Data (segment .data)

The fork()is one of the those system calls, which is called once, but returns twice!

- After fork()both the parent and the child are executing the same program.
- pid<0: the creation of a child process was unsuccessful.
- pid==0: to the newly created child process.
- pid>0: the process ID of the child process, to the parent.

