Processes in Linux

instigate-training-center.am

Agenda

- Process types
- Permissions
- Monitoring processes
- Links
- Creation/Ending
- State
- Times and timers
- Scheduling
- Process intercommunication

Instigate
Training Center

Process types

- Multi-user and multi-tasking
- Process types
  - init – parent of all processes
  - Interactive processes - initialized and controlled through a terminal session
  - Automatic processes - queued into a spooler area, where they wait to be executed on a FIFO basis
  - Daemons – server processes that run continuously

Permissions

- Unique identifier (pid)
- user (uid) identifier (chown)
- group (gid) identifiers (chgrp)
- Effective UID – SUID (chmod u+s)
  - The SUID permission makes a script to run as the user who is the owner of the script
- Effective GID – GUID (chmod g+s)
  - If a file is SGID, it will run with the privileges of the files group owner
- TTY – the terminal to which the process is connected
- Nice number – the friendliness of the process towards other processes (the priority of the process is calculated from nice numbers and recent CPU usage)

Monitoring processes

- ps - report a snapshot of the current processes
  - ps -ef - show every process on the system using standard syntax
- top - display Linux tasks
- pgrep - look up processes based on name and other attributes
  - pgrep -u student named - find the process ID of the named daemon run by student
- free - display amount of free and used memory in the system
- uptime - tell how long the system has been running

Links

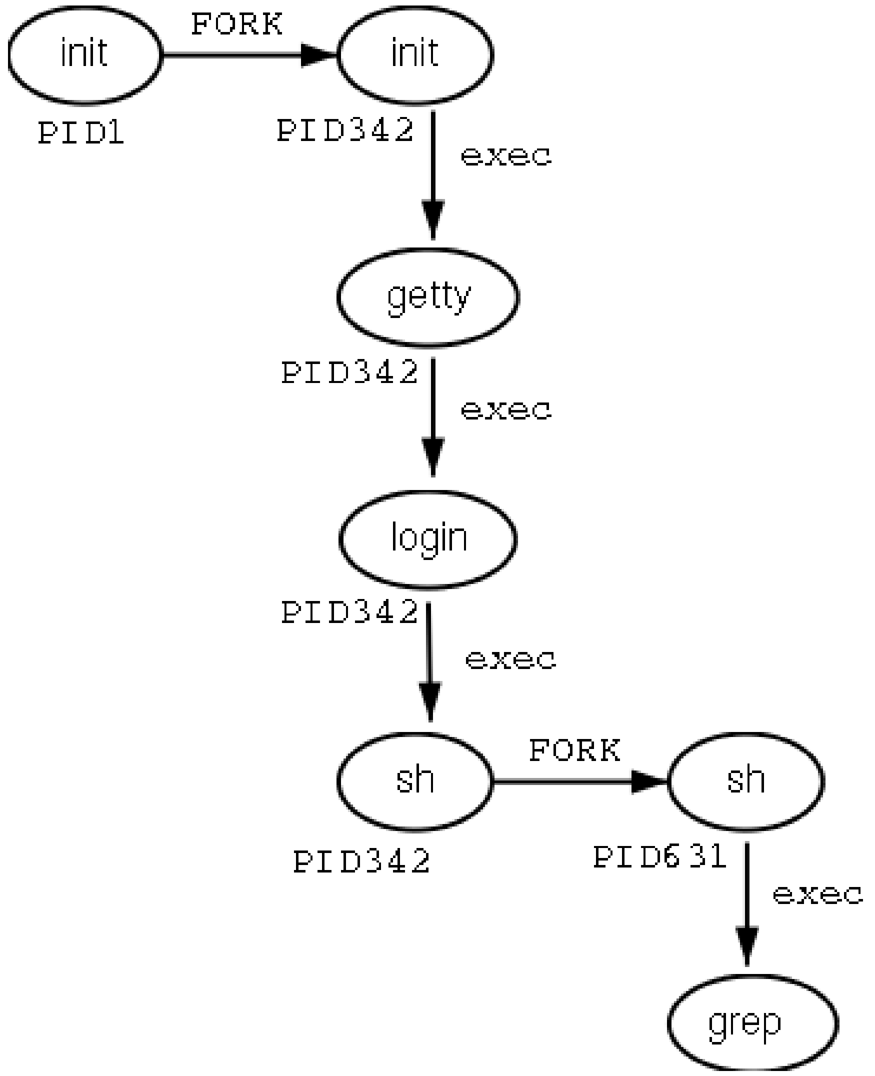
- Every process in the system, except the initial process has a parent process (ppid)
- Every process keeps info about parent, child and sibling (processes with the same parent) processes
- We can see the family relationship between running processes using pstree command

```
init(1)-+-crond(98)
        |-gpm(146)
        |-inetd(110)
        |-kerneld(18)
        |-kflushd(2)
        |-login(160)---bash(192)---emacs(225)
        |-lpd(121)
```

Process Creation/Ending

- New process creation by forking
- The address space of the child process is overwritten with the new data (exec)
- To send a signal to the process kill command can be used

• SIGTERM	15	Terminate the process in an orderly way.
• SIGINT	2	Interrupt the process. A process can ignore this signal.
• SIGKILL	9	Interrupt the process. A process can not ignore this signal.
• SIGHUP	1	For daemons: reread the configuration file.



Process state

- Running - the process is either running (it is the current process in the system) or it is ready to run (it is waiting to be assigned to one of the system's CPUs).
- Waiting - the process is waiting for an event or for a resource.
  - Interruptible - can be interrupted by signals
  - Uninterruptible - are waiting directly on hardware conditions and cannot be interrupted
- Stopped - the process has been stopped, usually by receiving a signal. A process that is being debugged can be in a stopped state.
- Zombie - this is a halted process, though its process data structure is not destroyed

Instigate
Training Center

Controlling processes

(part of) command	Meaning
regular_command	Runs this command in the foreground
command &	Run this command in the background (release the terminal)
jobs	Show commands running in the background
Ctrl+Z	Suspend (stop, but not quit) a process running in the foreground
Ctrl+C	Interrupt (terminate and quit) a process running in the foreground
bg	Reactivate a suspended program in the background
fg	Puts the job back in the foreground (e.g. fg %2)
kill, killall, pkill	End a process

Instigate
Training Center

Times and Timers

- The Kernel keeps track of a
  - processes creation time
  - CPU time that it consumes during its lifetime
- Processes can open/close files and in the internal process structure kernel stores
  - Descriptors for each open file
  - Pointers to 2 VFS inodes – process home directory and pwd (current directory)
- Processes have virtual memory and Kernel maps that virtual memory to physical memory
- When process is suspended registers, stacks, etc which are used during running are stored in the process data structure

Scheduling

- Processes have 2 types of policy
  - Round Robin (RR) – is preemptive – releases the CPU from long running jobs periodically based on timer interrupts
  - First In First Out (FIFO) – is nonpreemptive – process keeps running on a CPU until it is blocked/terminated
- The command *chrt* can be used for manipulating process policy
- User can alter the priority of the process using system calls and the *renice* command

Process intercommunication

- Inter-process communication
  - Signals
  - Pipes
  - Semaphores
  - Shared memory
  - Message queues
- Unnamed pipes
  - e.g. **ls -l | less**
  - e.g. **ps -ef | grep bash**
- Named pipes - can be created with **mkfifo** command
  - e.g. **mkfifo named_pipe**
  - e.g. **cat film.mp4 > named_pipe&; mplayer named_pipe;**

Instigate
Training Center