

CS-333 - Unix Operating System Fundamentals

Lecture 8 Controlling User Processes

Based on slides created by
Dr. Bangalore for the
Spring 2005 offering of
the course

Processes

- A process is a program in execution.
- Unix creates a process when executing the commands.
- Process is removed from the system when the command finishes its execution.
- Multiple process can be executed simultaneously by quickly switching from one process to the other.

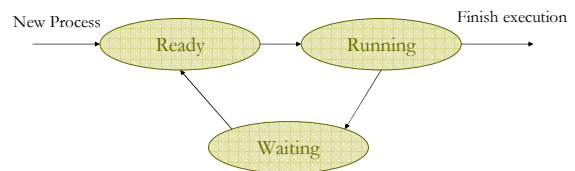
Code
Data
Stack
Process ID

10/20/2006

3

Process States

- A process can be in one of three states:
 - Ready
 - Running
 - Waiting: Pending I/O, child to exit, sleeping



10/20/2006

4

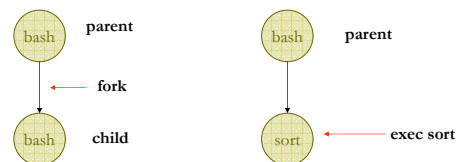
Process Creation

- A UNIX process can create another process using the fork system call.
- fork* creates an exact copy of the original process.
- Both processes continue execution.
- The code of the child process can be changed using the *exec* system call.
- The shell uses a combination of *fork* and *exec* to execute external commands.

10/20/2006

5

Example: bash executing sort



The parent (bash) waits for sort to finish and continue its execution.
If it did not wait, sort would become a zombie process once it completes execution.

10/20/2006

6

Department of Computer and Information Sciences UAB

Shell Details

Code: /bin/bash	fork	Code: /bin/bash
PID: 1234 PPID: 2222		PID: 7493 PPID: 1234
Local Variables: a = xyz		Local Variables:
Environment Variables: HOME = /home/afgane PATH = /usr/bin:.		Environment Variables: HOME = /home/afgane PATH = /usr/bin:.
Working Memory Input output error		Working Memory Input output error

10/20/2006 7

Department of Computer and Information Sciences UAB

Details about User Processes (I)

- To list user processes use "ps" (process statistics) command (on some systems you can also use "top")
- To obtain detailed information about all processes use "ps -ef"
- To list processes owned by you use "ps -u \$USER"
- To kill a particular process use "kill -9 PID" (to do so, you must either own the process or be ROOT)

10/20/2006 8

Department of Computer and Information Sciences UAB

Details about User Processes (II)

- To obtain a long listing of your processes use "ps -l"

```
$ ps -l
F S  UID  PID  PPID  C PRI NI  ADDR  SZ  WCHAN  TTY  TIME CMD
8 S  511 22817 22815  0  51 20  ?    309  ?  pts/1  0:00 bash
```

- F Flags
- S or STAT Process status code
- UID or USER Username of the process's owner
- PID Process ID number
- PPID ID number of the process's parent process
- C or CP CPU usage and scheduling information
- PRI Priority of the process
- NI nice value
- ADDR Memory address of the process C or CP CPU usage and scheduling information
- SZ Virtual memory usage
- WCHAN Memory address of the event the process is waiting for
- TTY or TTY Terminal associated with the process
- TIME Total CPU usage
- CMD Name of the process, including arguments, if any
- %CPU How much of the CPU the process is using
- %MEM How much memory the process is using
- VSZ Real memory usage
- START or STIME Time when the process started

10/20/2006 9

Department of Computer and Information Sciences UAB

Job Control Commands (I)

- When the shell executes a command it waits until the utility completes.
- If we do not like to wait, we can instruct the shell to execute the job in background by adding '&' at the end of the command.
 - Example: `sleep 10 &` - returns something like `[1] 14832`
- A *Job* is a process that is not running in the foreground and is accessible only at the terminal with which it's associated.

10/20/2006 10

Department of Computer and Information Sciences UAB

Job Control Commands (II)

- Shell may wait for a process to terminate using `wait` command.
- Example:


```
$ (sleep 30; echo done 1) &
24193
$ (sleep 30; echo done 2) &
24195
$ echo done 3; wait; echo done 4
done 3
done 1
done 2
done 4
$
```

10/20/2006 11

Department of Computer and Information Sciences UAB

Job Control Commands (III)

- To list all jobs background and stopped jobs use the command "jobs"
- To put suspended job into background use "bg %job_num"
- To bring a background job into foreground use "fg %job_num"
- To kill a job use "kill %job_num"

10/20/2006 12

Job Control Commands (IV)

- ❑ To suspend current job, press *CTRL-Z*
- ❑ To kill a running job with interrupt signal, press *CTRL-C*
- ❑ To kill a running job with quit signal, *CTRL-*
- ❑ *Unix Daemons* are system processes running in the background. e.j., *lpd* → printer daemon.