

# CS-344 - 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 finish its execution.
- ❑ Multiple process can be executed simultaneously by quickly switching from one process to the other.

---

---

---

---

---

---

---

---

Department of Computer and Information Sciences UAB

## Process States

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

```

    graph LR
      NewProcess[New Process] --> Ready((Ready))
      Ready --> Running((Running))
      Running --> Finish[Finish execution]
      Running --> Waiting((Waiting))
      Waiting --> Ready
  
```

10/28/2005 4

---

---

---

---

---

---

---

---

Department of Computer and Information Sciences UAB

## 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.

10/28/2005 5

---

---

---

---

---

---

---

---

Department of Computer and Information Sciences UAB

## Example: bash executing sort

```

    graph TD
      subgraph Forking
        P1((bash parent)) -- fork --> C1((bash child))
      end
      subgraph Executing
        P2((bash parent)) -- exec sort --> C2((sort))
      end
  
```

The parent (bash) waits for sort to finish and continue its execution.

10/28/2005 6

---

---

---

---

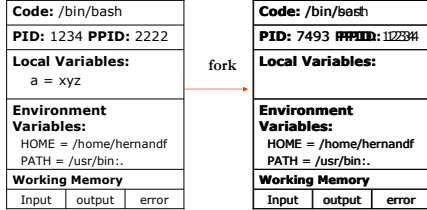
---

---

---

---

## Shell Details



10/28/2005

7

---

---

---

---

---

---

---

---

---

---

## Details about User Processes

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

```
$ ps -l
FS 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
```

- ❑ To kill a particular process use "kill -9 PID"

10/28/2005

8

---

---

---

---

---

---

---

---

---

---

## 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.

10/28/2005

9

---

---

---

---

---

---

---

---

---

---

## Job Control Commands (II)

- ❑ 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`

10/28/2005

10

---

---

---

---

---

---

---

---

## Job Control Commands (III)

- ❑ 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.

10/28/2005

11

---

---

---

---

---

---

---

---