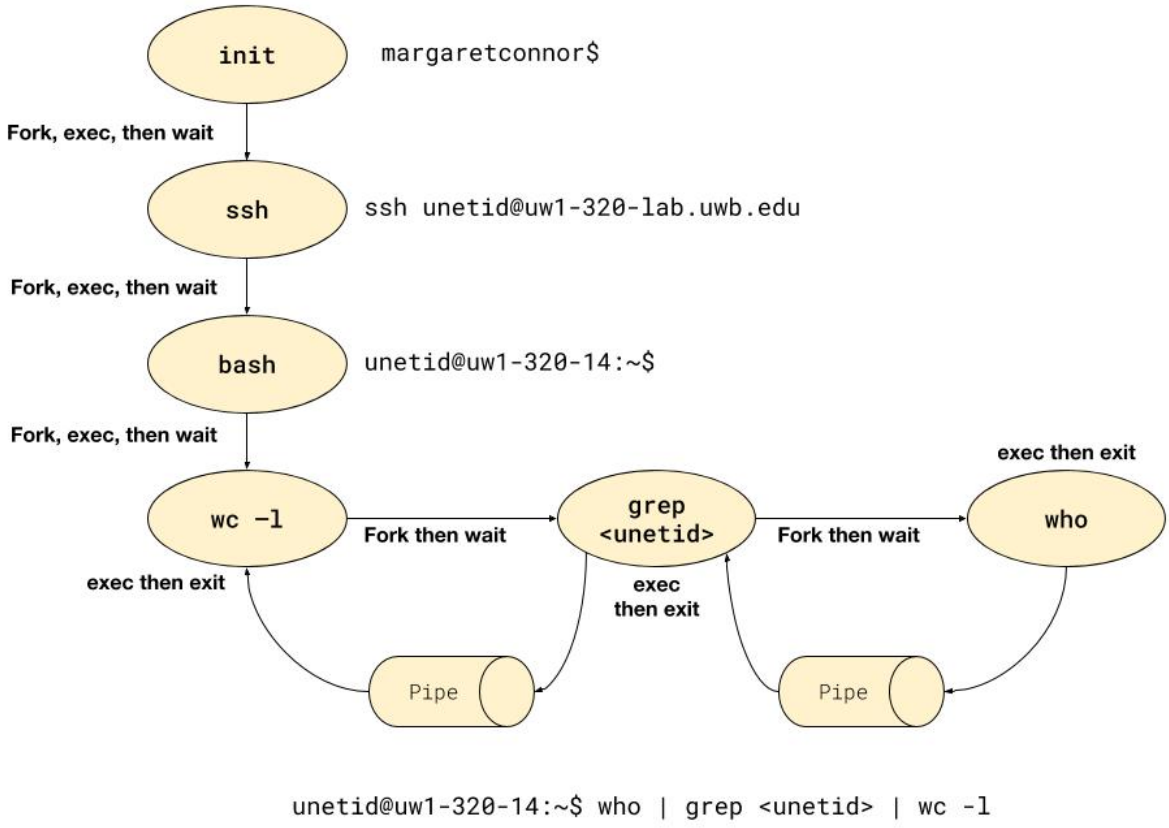
Margaret Connor
CSS 430

In Unix, the first process is called init. All the others are descendants of “init”. The init process spawns a sshd process that detects a new secure ssh requested connection (WKPort 22). Upon a new connection, sshd spawns a login process that then loads a shell on it when a user successfully logs into the system. Now, assume that the user types

who | grep <uwnetid> | wc –l

Draw a process tree from init to those three commands. Add fork, exec, wait, and pipe system calls between any two processes affecting each other.



“If the *shell* receives a sequence of commands, each concatenated with "|", it must recursively create children whose number is the same as that of commands. Which child executes which command is a kind of tricky. The farthest offspring will execute

the first command, the grand child will execute the 2nd last, and the child will execute the last command. Their standard input and output must be redirected accordingly using the pipe and dup2 system calls."

3.2 Including the initial parent process, how many processes are created by the program shown in Figure 3.31?

Including the initial parent process I belive 8 processes are created by the program.

3.9 Describe the actions taken by a kernel to context-switch between Processes.

The kernal performs a state save which saves the contests of current process to it's PCB (register) and then performs a state restor which loads in the next process PCB. The PCB includes the values of the CPU register, process state, and memory management information, during state saves and restores all this data need to be handled appropriately to allow the kernel to switch between processes.

3.11 Explain the role of the init process on UNIX and Linux systems in regard to process termination.

In Unix and Linux the init process serves as the new parent of orphaned processes. This occurs when the original parent process that spawned the to-be-orphaned processes terminates without calling wait(). The init periodically calls wait() which allows the orphaned processes to release their process identifies and their process table entry.

3.14 Using the program in Figure 3.34, identify the values of pid at lines A, B, C, and D. (Assume that the actual pids of the parent and child are 2600 and 2603, respectively.)

Assuming the parent process has a pid of 2600 and the child process has a pid of 2603 the program would produce

child: pid = 0
child: pid1 = 2603
parent: pid = 2603
parent: pid1 = 2600