Homework 2(Due by October 14,noon)-10 points

Misha Ward

Part 1:

2.6 - What system calls have to be executed by a command interpreter or shell in order to start a new process?

A fork() in UNIX/Linux or CreateProcess() in Windows is the command to use to start a new process. You would also need a exec system call to execute the next system calls. Since fork() copies the current process, exec allows that child process to become another program.

2.21 - What is the main advantage of the microkernel approach to system design? How do user programs and system services interact in a microkernel architecture? What are the disadvantages of using the microkernel approach?

Main advantages:

- Provide minimal process and memory management
- Allow easy communication
- Makes expanding the OS easier
- New services don't modify the kernel
- Operations are more secure
- Overall more reliable operating system

User programs interact with services with the use of message passing. Message passing is when systems communicate indirectly by exchanging messages with the microkernel.

Disadvantages:

- Microkernels can suffer from increased system function overhead.

2.22 - What are the advantages of using loadable kernel modules?

Loadable kernel modules are advantageous core components of the kernel link to additional services via modules. These links are either done at boot time or during run time of the system. These links are implemented dynamically and much more preferable than adding new features to the kernel directly. Loadable kernels are more efficient as modules do not need messages in order to communicate.

Part 2:

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.

Answer:

Homework 2(Due by October 14,noon)-10 points

Misha Ward

```
Init (wait + fork)  ->
        sshd (exec + wait + fork) ->
                bash (wait + fork) ->
                         wc -l  (fork + wait)->
                                grep (pipe) (exec)  (wait) (fork) ->
                                        Shell who (pipe + exec + exit)
```

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

8 processes, start with first fork() which means there are 2 processes, then fork() again to 4 processes, then fork() once more to 8 processes.

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

When a context switch occurs, the kernel:

1. saves the context of old process in PCB
2. loads the saved context of the new process scheduled to run

3.11 - explain the role of the init process on UNIX and LINUX systems in regard to process termination.

In UNIX or LINUX, a parent process without wait() can cause the child process to become “orphaned”. However, the child cannot be alone so the OS reassigns the child to the init process as the new parent. At this point, the init process invokes wait to ensure that the child process completes and freeing up those resources.

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.)

Ran program, received output similar to this but modified values:

```
parent: pid = 2603  C
parent: pid1 = 2600  D
child: pid = 0  A
child: pid1 = 2603  B
```