Writing Topic 1: Processes

Rhea Mae V. Edwards

CS 444
Spring 2017

When it comes to the implementations of processes, threads, and CPU scheduling across a variety of operating systems, such as Windows, FreeBSD, and Linux, there are many similarities and differences that exist when they are compared to one another. This report with focus on comparing the implementation of Linux to each of the Windows operating system and the FreeBSD operating system.

Processes

A process can be defined as the flow at which a task is used during a program. Processes have states, at first initializing the process itself, then put into a ready queue, waiting to be ran. While being in the running state, is when the process ends naturally by exiting or somehow blocked, for example by being killed by an external process. If a process is blocked, the only way a process can move out of that state, is to be moved back into the ready queue, which it is then can only be moved forward back to the running state. Another characteristic of a process' state diagram, is that a process can cycle through the running state and the ready queue.

Types of processes that can exist within an operating system are parent processes, child processes, orphaned processes, and zombie processes. Where child processes are created or forked from parent processes, and how orphaned processes are child processes that have been terminated where they have not being removed completely and properly and where the parent processes are still well running. Which then a zombie process is formed from an orphaned process that still has not been properly removed, and the parent process somehow ended without properly working with the orphaned process.

There are two ways a process can be processed, which are CPU bound, usually being faster, or usually I/O bound, being slower. The way these processes are used differ between various types of operating systems where some characteristics can still remain the same.

Windows

The Windows operating system runs by multiprocessing, being considers a CPU bounded process used throughout the operating system itself [1].

The Linux operating system supports real time processes, which work fairly quickly being considered as a CPU bound way of processing processes, causing similarities to the Windows operating system when compared [2]. Additional similarities and differences when compared to Linux, and why they exist.

FreeBSD

FreeBSD is a multi-tasking operating system, which is considered as a CPU bounded process for processing processes [3].

Linux implements real time processes, similar to FreeBSD, which work fairly quickly being considered as a CPU bound way of processing processes [2]. Additional similarities and differences when compared to Linux, and why they exist.

Threads

Threads are the pieces of executable program instructions that consist of a process. The ways threads are purposely used can differ and also hold similarities throughout various operating systems.

Windows

The Windows operating system implements threads through the method of multithreading [1].

Based on Linux's way of processing processes, relating to the implementation of threads throughout the operating system, Linux is similar to the way Windows implements multithreading with its threads within a process [2]. Additional similarities and differences when compared to Linux, and why they exist.

FreeBSD

FreeBSD being a multi-tasking operating system, characterizes FreeBSD implementation of threads in a multithreading fashion [3].

The FreeBSD is similar to Linux's way of processing threads, when analyzing the comparison with FreeBSD's implementation of multithreading and Linux's implementation of multiprocessing [2]. Additional similarities and differences when compared to Linux, and why they exist.

CPU Scheduling

CPU scheduling is the way a process is conducted by the planning and execution of the threads within a process. There are two different types of schedulers, low level schedulers and high level schedulers. Low level schedulers run the queue per CPU, select threads from the highest priority queue, and runs a round robin system for a quantum per thread. High level schedulers are primarily for SMP, where there is processor affinity, equidistribution, and multi-core and multi-socket implementation, being a O(1), constant time scheduler.

Various operating system implement their CPU schedulers differently which can also hold similar characteristics.

Windows

The Windows is a priority-driven preemptive scheduler, which implements CPU Scheduling as a low level type of scheduler [1].

The Linux operating system uses pre-emptive scheduling, which is considered as a high level scheduler, differing from the Windows operating system that implements a high level scheduler [2]. Additional similarities and differences when compared to Linux, and why they exist.

FreeBSD

FreeBSD implements its CPU scheduler in a multitasking environment, which would be considered as a high level type of scheduler [4].

The Linux operating system uses pre-emptive scheduling, which is considered as a high level scheduler, similar to the FreeBSD operating system [2]. Additional similarities and differences when compared to Linux, and why they exist.

Bibliography

[1] W. Stallings. (2005) *The Windows Operating System* [Online]. Available:
`http://avellano.fis.usal.es/~lalonsoamp_inf/windows.pdf`

[2] D. Rusling. (1999) *Chapter 4 Processes* [Online]. Available:
`http://www.tldp.org/LDP/tlk/kernel/processes.html`

[3] *3.8. Processes and Daemons* [Online]. Available:
`https://www.freebsd.org/doc/handbook/basics-processes.html`

[4] *2.4. Process Management* [Online]. Available:
`https://www.freebsd.org/doc/en/books/design-44bsd/overview-process-management.html`