Writing 1

A process is defined as: ‘a series of actions or steps taken in order to achieve a particular end’[-1].

Processes within FreeBSD, Linux, and Windows all share the same conceptual identity of being at the center of a program, but all three of these platforms are designed and managed through a few vastly different approaches. For instance, each of the three systems use different default managing systems, more colloquially referred to as schedulers, as well as each platform having different management systems, such as ULE or CFS. Processes are the foundation of any operating systems, and as such play an important role in creating a usable environment for the user to interface with the machine, and complete a given task. Within this Assignment, We will be comparing and contrasting FreeBSD and Windows to the Linux OS, to showcase features and mechanisms that each system possesses.

FreeBSD to Linux

FreeBSD and Linux both share a number of similarities to each other in process structure, this may be due to them both being relatives of the unix platform. Both of the systems use a monolithic kernel, which means all processes stem from one origin point during the boot procedure[cite1]. This means that FreeBSD will have a number of very similar structures such as the parent-child-zombie[1] relationship, as well as having a number of POSIX functions that are usable on either platform.

While they may share a similar origin, both of these platforms have Unique characteristics that the other does not share. One example being that when a child is forked off of the parent process, it will inherit its parent’s priority, but can lower the priority as CPU time is assigned. This is due to FreeBSD’s unique ULE scheduler. Unlike the CFS scheduler, ULE (Which is named after the last three letters of ‘Schedule’) will favor shorter running and blocking scripts[6]. Priority is effectively assigned to the fastest process to execute, however, a program requires longer and longer segments of processing time, its priority steadily gets decreased. Linux, on the other hand, uses the CFS scheduler, which stands for ‘Completely Fair Scheduler’ which uses a system of ‘fairness’ for each process to run. This system, does not apply the dynamic aspect of a task taking too long, rather, It appears to keep an internal tree of who can use what when, and does not adjust.[2]

Windows to Linux

Processes between Linux and Windows have a number of differences, for example, Linux will have the process contain the relationships it has to any other processes, when this is traced back to its origin, it finds a root node that fathers all of the processes used within the hierarchy. Within Linux each point of this hierarchy can possess one of three states, parent, child, or zombie[3]. Windows approaches this issue in a different manner, where it does not use a parent-child relationship, instead allowing each to run independently of each other. This allows process that are orphaned to not become zombies, and continue running without issue. As a result of this method, creation of a new process in windows is much more costly than Linux, because the system now needs to keep track of process information to prevent the zombie occuring.

For as different as the platforms operate with children, Linux and Windows share some strong similarities under the hood. Notably the Windows scheduling system and the CFS system Linux uses both possess a queueing system based on priority in order to manage the system, however, they have different approaches to implementing these systems. Linux comes with a priority system that seems to range from negative twenty to Nineteen, where a task is given a lower number if it has a higher priority[4]. Windows has a very similar system to this for its scheduler, instead using a queueing system that begins at the number zero and goes to thirty one. For windows, the numbers are also divided up into distinct categories, the Windows kernel, and the windows API[5].

Conclusion

Overall, the processes from Windows and FreeBSD both have a lot of strong similarities to those that exist in Linux, but they also differ in some key ways. For example, FreeBSD contains a scheduling algorithm that that is unique to it, the algorithm allows it to reassign all its priorities as a process receives CPU time. Similarities between the two stem from being distant cousins, as Linux is a distant derivative of the Unix kernel that FreeBSD uses, they have nearly identical process structure, where the only differences appear to be in the algorithmic implementations. An aspect unique to Windows, is that its processes receive different priority ranges, as well as receiving the process structure at the cost of the creation time due to bookkeeping. This tradeoff allows orphaned child processes to run independent of their parent, allowing you to either not have, or kill the parent process. Windows and linux are comparable in the concepts of their schedulers, both require fixed priorities, but also allow dynamic time slices to the CPU.Overall, the compared systems, FreeBSD, Windows, and Linux, appear to operate a lot more alike on the base kernel level than one may believe. With big differences stemming from their different approaches to child processes, schedules.

bib info:

<https://shankaraman.wordpress.com/tag/differences-between-windows-and-linux-process/> [3]

<https://idea.popcount.org/2012-12-11-linux-process-states/> [1]

<https://www.ibm.com/developerworks/linux/library/l-completely-fair-scheduler/> [2]

<https://www.ostechnix.com/change-priority-process-linux/> [4]

<https://msdn.microsoft.com/en-us/library/windows/desktop/ms685100(v=vs.85).aspx> [5]

<http://www.informit.com/articles/article.aspx?p=2249436&seqNum=4> [6]