Operating System Feature Comparison: Processes and Acheduling

by Sinan Topkaya

CS 444 - Spring 2016

Abstract: This paper will examine processes, threads and CPU scheduling for Windows and FreeBSD operating systems. It will mainly be about how these operating systems implement them and how is it different compared to Linux. Specifically, it will answer the questions: How do they differ? How are they the same? and Why do I think these similarities or differences exist?

Writing Assignment - Processes and Scheduling Comparison

Windows

Each Windows process is reptesented by an executive process(EPROCESS) structure. EPROCESS con tains and points to a number of other related data structures. Likewise if each process has o ne or more threads, each of them are repsented by and executive thread(ETHREAD) structure. To understand how Windows implements Processes and Threads we have to understand the structure EPROCESS and ETHREAD.

Processes: In this section I will write about how various data strctures involved in process state manipulation and management. Then I will write about how and when those data strcutures are created and filled out, while creating or terminating processes.

The EPROCESS and most of its data structures exist in system address space, except for process environment block(PEB), which exist in the process address space. The reason for that is because PEB contains information accessed by user-mode code. For each process that is executing Win32 program, the Win32 subsystem process (Csrss) maintains a parallel structure called the `CSR_PROCESS`

Threads:

CPU Scheduling: My implementation of the concurrency problem was done in C language. I have used Mersenne Twister to generate my random numbers. The problem has some contraints such as, while an item is being added or removed from the buffer, the buffer must be in an inconsistent state, I have acheived this through locking my the buffer for the specific thread that has to use it. Also if a consumer thread arrives while buffer is empty, it has to clock until producer adds a new item, I have achieved this by creating a checkempty funtion, this function check if the buffer is empty, and initalizes it to 1 or in other words true if so. Another constraint was that a producer thread should be able to put more items if the buffer is full, I have achieved this by using the same function but in !checkempty format. I have used getrandinit32 funtion for creating my random numbers.

FreeBSD

Processes:

Threads:

CPU Scheduling:

Answer to the questions

Windows

How is it different compared to Linux?:

How is it similar to Linux?:

Why do I think these similarities or differences exist?:

FreeBSD

How is it didderent compared to Linux?:

How is it similar to Linux?:

Why do I think these similarities or differences exist?: