3-2 Journal: Process and Thread Scheduling

Christopher Gray

IT: 600: Operating Systems

Southern New Hampshire University

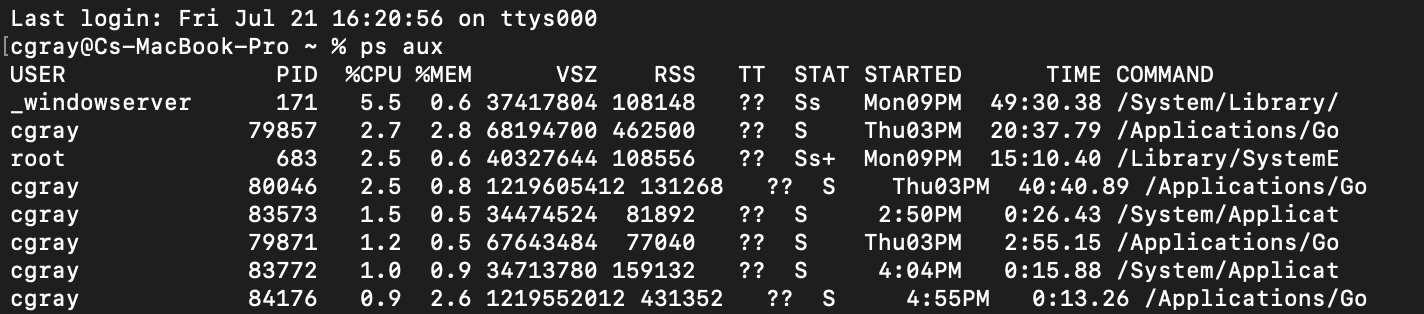
Professor Michael Gallaher

September 9, 2023

Journal Entry #2:

MacOS Monterey, like most modern operating systems, uses a priority-based scheduling algorithm. This means that processes are assigned priorities based on their characteristics, and the scheduler determines which process to run based on these priorities. Higher-priority processes are executed before lower-priority ones. macOS employs a variation of the priority-based algorithm to optimize system performance and responsiveness.

**ps aux command execution seen below:**



The running state for my computer is the Round Robin scheduling algorithm, in which, processes are assigned time slices and executed in a circular manner. The 'running' state in macOS Monterey indicates that the process is currently executing and utilizing its allocated time slice. The sleeping state is the First-Come, First-Serve (FCFS) scheduling algorithm, in which, processes are executed in the order they arrive. The 'sleeping' state in macOS Monterey refers to processes that are waiting for an event to occur before they can continue execution. These processes are similar to the waiting queue in FCFS, as they will be executed in the order they entered the sleeping state. Furthermore, The 'waiting' state in macOS Monterey corresponds to processes that are waiting for a specific resource or event to occur. In Priority-Based scheduling, processes with higher priority levels get access to resources faster than lower-priority ones. Consequently, processes in the 'waiting' state will be executed based on their priority. Moreover, macOS provides additional information with the 'ps aux' command, such as CPU and memory usage, user ownership, and process IDs (PIDs). The CPU and memory usage can help in analyzing the performance of processes, allowing us to identify potential bottlenecks and resource-intensive processes. The PID information is essential for identifying parent-child process relationships, which can be crucial when dealing with daemons or background processes.

The information obtained from the 'ps aux' command gave us insights into the status of threads and processes, enabling us to align our process management strategies with the requirements of our final project effectively. Whether supporting a server application, an interactive workstation, or a real-time sensor system, the appropriate scheduling algorithm should be chosen to optimize the system's performance and responsiveness.