CS 2301-01 Operating Systems 1

Instructor: Professor Ravichandran Purushothaman

Name: Anonymos

Written Assignment Unit 3

The U235 Nuclear Power Plant is committed to the well-being of its employees. Considering the demanding nature of the work environment, the plant has decided to provide a wearable device that tracks employees' health and relays this information to a smartphone app. The device is already prepared, and the mobile app will be developed by the app development team. My team is responsible for creating an operating system (OS) for this wearable device. This paper will discuss the method by which our device's CPU will choose process precedence, explaining why this method is optimal for the project.

**Method Selection**

For the wearable device's CPU to determine process precedence, we have selected the Completely Fair Scheduler (CFS) method, a well-regarded scheduling technique used in the Linux kernel. The CFS aims to allocate CPU time to processes in a manner that ensures fairness and efficiency, making it suitable for our project's requirements.

**Detailed Explanation and Rationale**

The CFS operates by maintaining a virtual runtime for each process. It tracks the amount of CPU time each process has consumed, ensuring that all processes receive an equitable share of CPU time. The scheduler selects the process with the least virtual runtime for execution, thus balancing the workload across all processes (Mauerer, 2010).

The key advantages of CFS for our wearable device are its low latency and high throughput. In a scenario where the primary function involves relaying health data and displaying information to the user, the responsiveness of the system is crucial. CFS minimizes the waiting time for I/O-bound processes, which are prevalent in our use case, thereby enhancing the real-time performance of the device (Bovet & Cesati, 2005).

Moreover, CFS's design aligns well with the constraints of a wearable device. Given the limited processing power and memory resources, the scheduler's simplicity and efficiency ensure that system overheads are minimized, contributing to the overall stability and performance of the device.

**Implementation Considerations**

Implementing CFS on our wearable device requires careful optimization to accommodate the hardware's resource constraints. This involves fine-tuning the scheduler's parameters to prioritize I/O operations effectively. Additionally, we must ensure that the virtual runtime calculations are lightweight to prevent any significant impact on the device's battery life and overall performance.

Another consideration is the integration of CFS with the real-time health monitoring requirements. This may involve implementing priority boosts for critical health alerts to ensure they are processed immediately, maintaining the reliability of the health tracking functionality.

In conclusion, adopting the Completely Fair Scheduler for the U235 Nuclear Power Plant's wearable device is a strategic decision that aligns with the project's goals of real-time health monitoring and efficient data relay. CFS's low latency, high throughput, and fairness make it an optimal choice, ensuring that the device performs reliably under the specified conditions. By leveraging this scheduling method, we can meet the client's requirements and contribute to the overall success of the project.

Word Count: 480

References

1. Bovet, D. P., & Cesati, M. (2005). *Understanding the Linux kernel (3rd ed.).* O'Reilly Media.
2. *Linux scheduler — The linux kernel documentation.* (n.d.). The Linux Kernel. Retrieved from https://www.kernel.org/doc/html/latest/scheduler/index.html
3. Mauerer, W. (2010). *Professional Linux kernel architecture.* Wrox.