Linux kernel-Completely Fair Scheduler

The Completely Fair Scheduler (CFS) is a process scheduler used in the Linux kernel. It was designed to replace the earlier O(1) scheduler and was first introduced in Linux kernel version 2.6.23.

Key Features of CFS:

1. **Time-slicing**: Each process is allocated a time slice, known as a quantum (also referred to as "tick" or "time slice"), which determines how long the process can execute before another process takes over.
2. **Fairness**: The scheduler ensures that each process has an equal chance of execution by dynamically adjusting the time slice based on the process's priority and other factors.
3. **Priority-based scheduling**: Processes are assigned a priority, with higher-priority processes executed first.
4. **Low latency**: CFS minimizes the overhead of context switching between processes.

How CFS Works:

1. The scheduler maintains a queue of all processes waiting to run (the "runqueue").
2. Each process is represented by an entry in the runqueue, which includes its current priority and time-slice allocation.
3. When a processor becomes available, the scheduler selects the next process from the top of the runqueue based on its priority and time-slice allocation.
4. The selected process executes for its allocated time slice (quantum).
5. After the quantum expires or the process voluntarily yields control back to the kernel, the scheduler updates the process's position in the runqueue based on its current priority.

Example:

Suppose we have three processes running simultaneously:

Process ID	Priority	Time Slice (Quantum)
P1	High (3)	100ms
P2	Medium (2)	50ms
P3	Low (1)	10ms

Initially, the runqueue is empty. When a processor becomes available, the scheduler selects the process with the highest priority, which is P1. P1 executes for its allocated time slice of 100ms.

After P1's quantum expires, the scheduler updates the runqueue by moving P2 to the top (since it has the next highest priority). P2 then executes for its allocated time slice of 50ms.

Finally, after P2's quantum expires, the scheduler moves P3 to the top and allocates its own small quantum. P3 executes until completion or voluntarily yields control back to the kernel.

Benefits of CFS:

1. **Improved fairness**: Processes are executed in a more predictable and fair manner.
2. **Reduced latency**: Context switching is minimized, resulting in lower latency between processes.
3. **Efficient use of resources**: The scheduler optimizes the allocation of time slices to minimize idle time.

Caveats:

1. **Increased overhead**: CFS introduces some additional overhead due to the dynamic updating of process priorities and time-slice allocations.
2. **Sensitivity to system load**: The effectiveness of CFS can degrade under high system loads or when there are many processes competing for resources.

Overall, the Completely Fair Scheduler is a sophisticated and efficient scheduling algorithm that provides good fairness, low latency, and resource utilization in Linux systems.

Curated by Brajesh Kumar