slides

Process Scheduling

Oslab Seminar

Vivek Sehgal – 00005022
Guided by Prof. R. K. Joshi

Kernel Overview

Every time the scheduler does the following:

- Kernel work.
- Current process.
- Process selection(Scheduling Algorithm).
- Switching processes.

Scheduling Algorithm

Priority

- In linux 2.4, priority of a process ranges from 0 to 139.
- 0-99 priority rangeis called the “static”. It is allocated to real-time tasks and never changed by the scheduler.
- 100-139 priority range is called the “dynamic”. It is for the conventional processes.
- The static priority is always higher than the dynamic priority - the real-time processes always have precedence over the conventinal processes.

Scheduling Algorithm(contd.)

Time Slice

- The higher a process's priority, the bigger timeslices it gets during one round of execution.

- In Linux 2.4, the minimum timeslice is 10msec, the default timeslice is 150msec, and the maximum timeslice is 200msec.

Scheduling Algorithm(contd.)

Data Structure : The most important datastructure relevant to a scheduler is the task structure. It includes:

- p->need-resched: this field is set if schedule() should be invoked at the 'next opportunity'.
- p->counter: number of clock ticks left to run in this scheduling slice, decremented by a timer.
- p->priority: the process' static priority, only changed through well-known system calls like nice(2).
- p->rt-priority: realtime priority

- p->policy:
  - SCHED-OTHER (traditional UNIX process)
  - SCHED-FIFO
  - SCHED-RR (round-robin).

  One can also OR SCHED-YIELD to any of these values to signify that the process decided to yield the CPU.

Schduling Algorithm(contd.)

How to choose a process for execution?

- The runqueue is examined and a goodness of each process that can be scheduled on this cpu is calculated. The process with highest goodness wins.

- The goodness for realtime processes is calculated by making their goodness very high (1000 + p->rt-priority).

- For non-realtime processes, the initial value of goodness is calculated such that the process is less likely to get CPU if it already had it for a while.

References

- Linux 2.4 Kernel Internals,
  http:" www.moses.uklinex.net/patches/lki.html
- The Linux Kernel,
  http://www.linuxdoc.org/LDP/tlk/tlk.html