Advanced Operating Systems
Report 5

37186305
航空宇宙工学専攻修士一年
荒居秀尚

2018 年 11 月 22 日

1 The difference between nice priorities and RT priorities

1.1 Description of processes

Every process can be categorized into three types.

Interactive process
Those processes which are categorized in this type always need to be interactive with the user. The biggest feature of this type is that processes need to wait long time for the action of the user such as keyboard input or mouse manipulation. These processes need to wake up immediately when the input comes, in order to prevent user from feeling slow. Examples of this type of processes are command shell, text editor, and GUI applications.

Batch process
This type of processes don't need the manipulation of the user, and they are often running as background processes. Responsiveness is not important for this type, so they are sometimes degraded their priorities by the scheduler. Example of this type of processes are compilers, search engine for the database, and scientific calculation.

Real Time process
The scheduling requirements are very strict for this type and they should not be disturbed by low priority processes. They need to function quickly and stably. Example of this type of processes are video or audio applications, robot control, and sensing program.

1.2 Scheduling policy

Each Linux process has its scheduling policy.Real Time processes have three options, SCHED_FIFO, SCHED_RR, and SCHED_DEADLINE.

SCHED_FIFO

FIFO is the abbreviation of ”First In First Out”.When the scheduler allocated CPU to the process which has this policy, the position of process descriptor in the execution queue would not change.If there aren’t any process which has higher priority than the current process (with this policy), the current process will not stop using the CPU even if there is a process which has the same priority as the current process.

SCHED_RR

RR is the abbreviation of ”Round Robbin”.When the scheduler allocated CPU to the process which has this policy, the position of process descriptor will be moved to the end of the execution queue.When the process run out of the allocated quantum, the process stop running and wait until the other processes which have the same priorities as the current one and have the policy SCHED_RR use one quantum.

SCHED_DEADLINE

Process with this policy has the deadline and the priority value is set to the highest automatically.

Interactive processes and batch processes may have three options for their policy, SCHED_OTHER or SCHED_NORMAL, SCHED_BATCH, and SCHED_IDLE.The user can set a value called ”nice value” for the processes with SCHED_OTHER / SCHED_NORMAL and SCHED_BATCH.This nice value is the essence of the ”nice priority”.

1.3 Scheduling for Real Time process

Every Real Time process has its own RT priority.This priority is a value between 1 and 99.The higher this value is, the lower its priority.The scheduler always pick the process which has the highest priority, which means that execution of those processes which has lower priority than the RT process are suppressed while RT processes are executed.

RT processes give over the execution right to another process only when the incident below happens.

- The process was preempted by another process which has higher real time priority.
- The process interrupted the execution and stopped running with the status of TASK_INTERRUPTIBLE or TASK_UNINTERRUPTIBLE.
- The process was interrupted and the status changed to TASK_STOPPED or TASK_TRACED, or the process was forced to exit and the status changed to EXIT_ZONBIE or EXIT_DEAD.
- The process called the system call sched_yield() and spontaneously give over the execution right.
- The process has the attribute of SCHED_RR and run out of the quantum.

Real Time process are always given higher priority than the normal (nice) priority.

1.4 Scheduling for Normal process with nice priority

Processes with nice priority use two value to calculate the quantum time.When the process spend all the given time slice, the execution of the process is preempted.The two value is, **static priority** and **bonus** value.With these values, Linux scheduler calculate dynamic priorities.Both static priority and dynamic priority can take a value between 100 and 139.The higher the value is, the lower the priority is.This is the same as the real time priority.Nice

value is a value between -20 and 19, which can be calculated with the formula $nice = static - 120$.The bonus value is decided from the past activity of the process.In detail, ”past activity” means average sleeping time.If average sleeping time of a process is more than 200 mili seconds, the process is judged to be an interactive process.The higher this bonus is, the high dynamic priority is.

1.5 Rescheduling for Real Time processes

When the current process is Real Time process with SCHED_FIFO policy, rescheduling is not needed.FIFO type Real Time processes won't be preempted by processes with lower priority.
If the current process is RR type Real Time process, the scheduler_tick() function decrease the time slice counter of the process and check whether the process has run out of its quantum.When the scheduler judged that RR type RT process has run out of the quantum, scheduler execute tasks to preempt the execution right from the current process if needed.The first task is to fill the time slice counter with another time slice.The scheduler calculate the quantum time from the process's static priority value.After that the scheduler set the first_time_slice flag to zero and set the TIF_NEED_RESCHED flag.The last task is to move the current process descriptor to the end of the execution queue of the processes with the same priority as current process's.

1.6 Rescheduling for normal processes

When a normal process has run out of its quantum, the scheduler_tick() function will do the tasks below.

1. call dequeue_task() function to delete the current process from this_rq -> active set.
2. set the TIF_NEED_RESCHED flag.
3. change the current process's dynamic priority.
4. calculate the quantum time and fill the time slice counter with basic quantum time slice.
5. set the expired_timestamp member to the current tick if the expired_timestamp is not set to zero.
6. insert the current process to the active processes set or expired processes set.
7. check if the remaining time slice is too long, if the time slice of the current process is remained.

1.7 summary

Real Time priority has the value between 0 and 99 and nice priority has the value between -20 and 19.With this value, the scheduler decide which process to run.Real Time processes always have higher priorities than normal processes, and cannot be preempted by those processes except for a few exceptions.
The biggest difference between the RT priority and nice priority is the way to calculate the basic quantum time.RT processes with FIFO policy do not have the idea of quantum time.RT processes with RR policy have quantum time, but that time slices are only calculated with the static priorities.On the other hand, normal processes with nice priority also have the quantum time, but that time slices are decided based on the dynamic priorities.Dynamic priority is calculated from static priority and the average sleeping time of the process.

2 reference

[1] Daniel P. Bovet, Marco Cesati. (2007). Understanding Linux Kernel, 3rd Edition. O'REILLY.