Assignment 7: Exploring the Linux Scheduler

Deadline: 10th October 2015, 9:00 pm

Goal: In this assignment, you will explore various scheduleing options available in Linux with the given test program.

Task:
You will run test programs on a single core environment with processor affinity(it determines the set of CPUs on which a process is eligible to run) for testing various Linux schedulers. The objective of this program is to test a program with various scheduling parameters.

Create a main program (process) which forks *k* child processes of type <longrun.c>. For simplicity, you can assume that *k* is a small number, typically less than or equal to 5. You will have to run each child process with different schedulers and any priority (if any). So, you will get from the user the following inputs: (1) Number of child processes, i.e. *k* (2) Scheduler for each of the child processes; (3) Priorities of the child processes (if they are relevant).

The set of schedulers available in linux are:

- SCHED_OTHER: the standard round-robin time-sharing policy;
- SCHED_BATCH: for "batch" style execution of processes; and
- SCHED_IDLE: for running very low priority background jobs.

The following "real-time" policies are also supported, for special time-critical applications that need precise control over the way in which runnable threads are selected for execution:

- SCHED_FIFO: a first-in, first-out policy; and
- SCHED_RR: a round-robin policy.

Processes scheduled under one of the real-time policies (SCHED_FIFO, SCHED_RR) have a sched_priority value in the range 1 (low) to 99 (high).

You have to prepare a report with the test cases.

Test case 1: *k = 2*
Process 1: Scheduler (FIFO) and Priority HIGHEST
Process 2: Scheduler (FIFO) and Priority LOWEST
Expected Result: ?

Test case 2: *k = 2*
Process 1: Scheduler (FIFO) and Priority HIGHEST
Process 2: Scheduler (FIFO) and Priority HIGHEST
Expected Result: ?

Test case 3: *k = 2*
Process 1: Scheduler (RR) and Priority HIGHEST
Process 2: Scheduler (RR) and Priority LOWEST
Expected Result: ?

Test case 4: *k = 2*
Process 1: Scheduler (RR) and Priority HIGHEST

Process 2: Scheduler (RR) and Priority HIGHEST
Expected Result: ?

Test case 5: *k = 3*
Process 1: Scheduler (SCHED_OTHER) and (static) Priority 0
Process 2: Scheduler (FIFO) and Priority LOWEST
Process 2: Scheduler (RR) and Priority LOWEST
Expected Result: ?

Test case 6: *k = 3*
Process 1: Scheduler (SCHED_OTHER) and Priority 0
Process 2: Scheduler (RR) and Priority LOWEST
Process 3: Scheduler (RR) and Priority HIGHEST
Expected Result: ?

Each of the above test cases be run for 3-4 times before you make any conclusions on the order of processes execution.

Input Parameters to the Program:

You have to take input from a file, 'input.txt' the above mentioned parameters: *k*, the scheduler for each child process, the priority of each child process.

Output of the execution:

Show the display of the execution of longrun.c by the various child processes.

Deliverables:

You have to submit the following:

- A report consisting of the test cases as described above
- The source program named as <rollno.>-sched.c
- longrun.c, in case you have changed the code of the test program, named as <rollno.>-longrun.c
- a readme file

Create a zipped archive of all your files and name it as <rollno.>-assgn7.zip. Upload it on google classroom. Strictly follow this naming convention. Otherwise, your assignment will not be evaluated.

Deadline: 10th October 2015, 9:00 pm

Note: The This assignment is similar to the scheduling assignment given last year as a part of the OS assignment. The test program longrun.c is the same test program as last year. Feel free to discuss with your seniors, but please do not copy their code.

Reading Materials and other Useful Links:

http://idak.gop.edu.tr/esmeray/UnderStandingKernel.pdf[chapter07]
http://man7.org/linux/manpages/man7/sched.7.html
http://man7.org/linux/man-pages/man2/sched_setaffinity.2.html
http://www.linuxperts.org/manual/linux/sched_setscheduler/2
http://www.csl.mtu.edu/cs4411.ck/www/NOTES/process/fork/exec.html

Appendix: System calls related to scheduling

System call	Description
nice( )	Change the static priority of a conventional process
getpriority( )	Get the maximum static priority of a group of conventional processes
setpriority( )	Set the static priority of a group of conventional processes
sched_getscheduler( )	Get the scheduling policy of a process
sched_setscheduler( )	Set the scheduling policy and the real-time priority of a process
sched_getparam( )	Get the real-time priority of a process
sched_setparam( )	Set the real-time priority of a process
sched_yield( )	Relinquish the processor voluntarily without blocking
sched_get_priority_min( )	Get the minimum real-time priority value for a policy
sched_get_priority_max( )	Get the maximum real-time priority value for a policy
sched_rr_get_interval( )	Get the time quantum value for the Round Robin policy
sched_setaffinity()	Set the CPU affinity mask of a process
sched_getaffinity()	Get the CPU affinity mask of a process