Operating Systems - Winter 2018

Sambuddho Chakravarty

February 18, 2018

Assignment 2

Due date: March 11, 2018. Time: 23:59 Hrs.

1 Multi-user chat system (Total points: 45)

As presented in the class, you are required to design and implement a multi-user chat system that is like a real-time instant message board system, much like IRC. The users should be able to talk to one another and to everyone else. The system needs to make sure which message is meant for whom – for individual user or for everyone. The message meant for everyone should be flashed on the system for everyone to view. The messages meant for individual users need to be handled by the individual users themselves.

You may user popular Linux IPC mechanism for the same. For now, please ignore synchronization/race-condition prevention.

What To Submit

- Program source code with Makefile
- Write-up describing the following:
  - Description of your code and how you implemented the function – the logical and implementation details.
  - Description of how to compile and test the program
  - The inputs the user should give.
  - Expected output (and how to interpret it).
  - Error values and how to interpret them.

Grading Rubric

- Successful compilation your the program – 10 points.
- Correct functioning of the chat system for multi-users – 20 points.
- Correct handling of input errors (atleast two different types of errors should be handled) – 10 points.
- Description of the systems, test cases *etc.* – 5 points.

2 Modifying CFS Scheduler (Total points: 45)

We spoke about the Linux CFS scheduler in class. It is however a non-realtime scheduler, with non-realtime guarantees being given to processes.

As a part of this second assignment you need to add a soft real-time requirement to a process – say something like each process, that requires soft real-time guarantees must receive atleast x units of time-slice. Every time the scheduler is called, you need to check if the real-time guarantees of process with soft-realtime requirements are being met or not. You would give higher priority to a processes soft-realtime requirement compared to the *vruntime* that is normally considered.

You would thus need to modify the scheduler in such a way that when everytime a process is selected through the RB-tree, you you need to compare to all other process with soft-realtime requirements and see which one of those require the CPU more urgently than the one selected through the regular RB-tree. The one that urgently requires CPU time need to be scheduled before the one selected through CFS.

You would additionally require a system call to modify the tasks soft-realtime requirements – say `rtnice`, which takes argument the PID and the realtime guarantees needed. `rtnice` can have a corresponding user program with the same name which should actually do the system call `rtnice` to perform the said operation with the supplied PID.

To test the correct functionality of the modified scheduler, you need to run tasks requiring realtime guarantees. You would need to set the guarantees through the `rtnice` program. You would need to show execution time of the process with and without the soft realtime guarantees. This should demonstrate to you if the modified scheduler is functioning or not.

What To Submit

- You need to submit the `diff`, of the originally downloaded kernel source tree and the one with your changes. We would patch that `diff` with our copy of the original source tree and compile the kernel.
- Write-up describing the following:
  - Description of your code and how you implemented the function – the logical and implementation detaisl.
  - The inputs the user should give.
  - Expected output (and how to interpret it).
  - Error values and how to interpret them.
- Sample scripts/test cases on how to test the system.

Grading Rubric

- Successful compilation your diff against the base kernel source – 10 points.
- Correct functioning of your modified scheduler along with the system call, testable through the supplied script(s) and/or test cases – 20 points.

- Correct handling of input errors (atleast two different types of errors should be handled) – 10 points.
- Description of the systems, how to test the system *etc.* – 5 points.

Late Submission Policy

- Submitted on or before March 11, 2018 (23:59 hrs) – No points deducted.
- Submitted after March 11, 2018 but on or before March 13, 2018 (23:59 hrs) – 5 points deducted.
- Submitted after March 13, 2018 but on or before March 15, 2018 (23:59 hrs) – 15 points deducted.
- Submitted after March 15, 2018 – no points shall be awarded.