Assignment 5

Customising the Xv6 OS

Course : Operating Systems (Monsoon 2018)

Deadline : 5th November, Monday, 23:59

The goal of this assignment is to understand the working of a toy OS - xv6 and implement few system calls for the same.

The following are the specifications for the assignment.

Specification 1: Simple Command

Implement 'invertcase' command in xv6. This command takes string arguments and prints out the case inverted string.

Eg:

$ invertcase hello World!
HELLO wORLD!

Specification 2: Priority Based Scheduling

The default scheduler of xv6 is a round-robin based scheduler. Implement a priority-based scheduler to replace the default scheduler.

A priority based scheduler selects the process with highest priority for execution. In case two or more processes have same priority, we choose them in a round robin fashion. The priority of a process can be in the range [0,100]. Smaller value will represent higher priority. Set the default priority of a process as 60.

Implement the syscalls - **ps** and **set_priority.**

- **-** where ps prints out the currently running processes names, pids and their priorities.
- set_priority is used to change the priority of a process.

int set_priority(int) - is the function declaration to be used while implementing the syscall.

Submit a report with a small example which demonstrates the working of your scheduler, the report should include comparison of your current (priority based) scheduling policy and the original round robin approach.

Hint: Modify the proc structure for storing the priority of every process.

Write a sample benchmark program (make sure it runs for at least 20 secs on your laptop) which can be used to compare the performances of the scheduling algorithms and demonstrate using that program during the evals.

Specification 3: System call Tracing

Modify the xv6 kernel to print out a line for each system call invocation.

When you're done, you should see output that looks something like this when booting xv6:

```
...
fork -> 2
exec -> 0
open -> 3
close -> 0
write -> 1
write -> 1
```

The above should work even for regular commands, and display appropriate call tracing output.

General Notes:

1. The xv6 OS base code can be downloaded from https://github.com/mit-pdos/xv6-public
2. Whenever you add new files do not forget to add them to the Makefile so that they get included in the build.
3. Documentation - https://pdos.csail.mit.edu/6.828/2011/xv6/book-rev6.pdf
4. File Docs - http://www.cse.iitd.ernet.in/~sbansal/os/previous_years/2011/xv6_html/files.html

Submission Guidelines:

- Upload format rollnum_assgn5.tar.gz.
- **Make sure you write a Makefile for compiling all your code (with appropriate flags and linker options)**