Exam in EDA050 Operating Systems

August 25, 2010, 8-13

Inga hjälpmedel!

Examinator: Jonas Skeppstedt

30 out of 60p are needed to pass the exam.

1. (10p) Suppose you have a file which normal users should not be allowed to edit — that is, it should be write protected. However, users should be allowed to run a program you have written which may change the contents of the file. How can you achieve that on UNIX?

   Answer: *See slides on SETUID programs.*

2. (20p) Compare a traditional UNIX file system of your choice (eg EXT2) with the Berkeley Log-Structured File System and EXT3. What has motivated the developments seen? What is new in EXT4?

   Answer: *See slides for a description of Berkeley LFS. The motivations include the needs for faster I/O to match the performance improvements achieved with the RISC processor revolution, and a need for faster recovery due to the disks became much larger than earlier. The point where performance improvements are needed are for disk writes — since file reads usually are from a cache in RAM.*

   New in EXT4 is support for larger file systems and better support for multimedia — the latter through allocation of larger blocks, called extents.

3. (10p) Describe a virtual memory system in which it is the kernel which takes care of TLB-faults.

   Answer: *See slides.*

4. (10p) Explain the following about UNIX pipes.

(a) (2p) What is the purpose of having pipes in UNIX?

Answer: *Making it possible to use the output of one process as input to another process in a simple and programmer friendly manner.*

(b) (4p) When a shell runs commands connected using pipes, why is it important that the shell process closes its file descriptors for writing to the pipes?

Answer: *Otherwise the kernel cannot know it should give EOF to the process reading from the pipe.*

(c) (4p) Why is the concept of process groups useful with pipes?

Answer: *If we for instance want to kill, stop, or resume the entire pipeline, it's sufficient (for the shell) to make one* `kill` *system call with a negative process id so that it is delivered to all processes in the group*

5. (10p) Show how a plain spinlock using the test-and-set instruction should be implemented using two loops. What is a ticket-based spinlock, what hardware feature does it require, and why is better than the plain spinlock?

Answer: *See slides for a simple spinlock based on test-and-set. A ticket-based spinlock requires an atomic fetch-and-increment instruction, and it is better than a simple spinlock since when the lock is released only one CPU will try to take the lock by writing to it. Compare this with the simpler spinlock in which all CPUs will write to the lock.*