Exam in EDA050 Operating Systems

May 27, 2010, 14-19

Inga hjälpmedel!

Examinator: Jonas Skeppstedt

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

1. (10p) Traditional UNIX File Systems

   (a) (2p) Why is the system call to remove a file from a directory called `unlink` and not `remove`?

   Answer: *The system call removes only the directory entry and not the file itself, unless it was the last directory entry referring to that file.*

   (b) (4p) Except for mount points, why can a normal UNIX directory not contain files stored in a file system (hard disk partition) different than the directory itself is stored in?

   Answer: *The directory contains mappings from file names to inode numbers, and in particular there is no attribute which specifies a partition.*

   (c) (2p) How can the kernel know whether an executable file is a shell script or, for instance, an ELF executable file?

   Answer: *By looking at the first few bytes, which are called the magic number. Scripts start with* `#!`path *while the first four bytes of an ELF executable are* `0x7f`*,* `'E'`*,* `'L'`*, and* `'F'`*.*

   (d) (4p) Which is the most frequent file operation and which is the most frequent disk access operation and why are they not the same?

   Answer: *Reads are the most frequent file operation and writes are the most frequent disk operation and the reason is the high read hit rate of the buffer cache.*

2. (10p) Modern File Systems

   (a) (2p) The BSD Log-Structured File System (LFS) needs an inode map. Why?

   Answer: *A modified inode is written in the log and to find the current version of an inode the inode map is needed. An inode thus has no permanent location on the disk in LFS as on eg EXT2 and EXT3.*

(b) (2p) What does the cleaner process do with LFS?

Answer: *The purpose of the cleaner process is to perform garbage collection in order to create fresh segments that can be used for writing in the log. It does so as follows: the parts of a segment that contain dead data (overwritten data blocks of a file, or removed files) need no action. If a segment S_1 has live data (ie still in use) that and live data from other segments can be collected and copied to a segment S_2 in order to make S_1 reusable.*

(c) (3p) What is the key idea behind EXT3 as opposed to BSD LFS and EXT2?

Answer: *Instead of as in BSD LFS using an entire disk partition as a log, EXT3 keeps the EXT2 file system structure but extends it with a small log to which writes first a performed. Subsequently the data is copied to the normal disk blocks.*

(d) (3p) EXT4 supports fast accesses to large files. How?

Answer: *By using so called extents which are areas of size up to 128 MB consisting of consecutive disk blocks. I/O is faster with larger disk blocks.*

3. (20p) Virtual memory.

(a) (1p) In which decade was virtual memory invented?

Answer: *1956.*

(b) (4p) Translating virtual to physical page numbers for every user program memory access seems to significantly degrade performance. Why does it ***usually*** not?

Answer: *The TLB's cache translations and usually a translation can be found in a TLB in which case the translation costs no additional clock cycles. It is very important that the hit ratio in a TLB is very high, otherwise performance degrades quickly.*

(c) (2p) What is a TLB-fault and what does the kernel (if it is involved) do about it?

Answer: *A TLB-fault is an exception triggered when a translation was not cached in the TLB. The kernel will look up the page table entry, and replace some other translation in the TLB with the new one. If the page was not in RAM more processing is required.*

(d) (5p) What is a multilevel page table and what is its disadvantage in a 64-bit architecture when there is a TLB-fault?

Answer: *To look up a translation in a multilevel page table requires a fixed but relatively large (depending on the number of levels) number of memory accesses to find the page table entry. With a 64-bit address space the number of levels typically is larger than for a 32-bit address space.*

(e) (3p) When a physical page is replaced and used for another virtual page, how can the kernel find the previous owner of the physical page and why would it want to find it?

Answer: *The coremap, indexed by physical page number, contains a pointer to the owning page table entry, which must be updated so that its data can be found on the swap when needed in the future.*

(f) (3p) Describe the second-chance page replacement algorithm.

Answer: *The coremap array is searched and when a page table entry with reference bit zero is found, then that physical page is used. When the reference bit is one, it is instead set to zero. The search wraps around after the last page has been inspected.*

(g) (2p) What is a linear page table and where is it stored ?

Answer: *A linear page table is an array of page table entries and since it is so large, it is stored in virtual memory.*

4. (10p) Synchronization.

(a) (4p) How can false cycles be avoided in distributed deadlock detection when there is a central coordinator that detects cycles?

Answer: *The nodes maintain the partly resource allocation graphs which are sent to the central coordinator when requested. Since there can then be false cycles due to a deletion of an edge from a resource allocation graph may happen before an addition of an edge but may actually be observed by the coordinator in to have happended after the addition. To avoid false cycles, the local resource allocation graphs can contain a time stamp in both the head and tail of edge and when a nodes*

(b) (6p) What is a ticket-based spinlock and what hardware feature does it require? Which performance problem does this lock have, if any?

Answer: *Atomic fetch-and-increment is needed. A disadvantage with ticket-based spinlocks is that at an unlock, all waiting CPUs will fetch the counter telling whose turn it is, while it would be better that only the CPU which will get the lock should be informed.*

5. (10p) Scheduling

(a) (6p) What is priority inheritance and how does Solaris implement it? Does their implementation have any limitations, and in that case which?

Answer: *See slides.*

(b) (4p) What is affinity scheduling and what is its purpose?

Answer: *To schedule a thread on the same CPU as it was running last time in order to have some useful data in the cache left.*