Department of Computer Science and Engineering
Chalmers

Exam in Unix Internals EDA203/DIT681

DAY: 2009-05-26 TIME: 14.00-18.00 PLACE: V building
==========

Examiner: Arne Dahlberg

Questions during exam: Arne Dahlberg, 772 1705

Solutions: No solutions will be posted

Grading Policy: EDA203 3: 30-38, 4: 39-47, 5: 48-60
DIT681 G: 30-47, VG: 48-60

Aids during the exam:

- McKusick, Neville-Neil: The design and implementation of the FreeBSD operating system.
- An English language dictionary.

Instructions:

- Start answering each assignment on a new page; number the pages and use only one side of each sheet of paper.
- Justify all answers. Lack of justification can lead to loss of credit even if the answer might be correct.
- No references to the text book is allowed and copying of text from the book is not allowed.
- If you make any assumptions in answering any item, do not forget to clearly state what you assume.
- Write clearly! If I cannot read your solution, I will assume that it is wrong.

Problem 1 (10p)

When doing voluntary process switches, the kernel use the subroutines sleep and wakeup.

a. Describe the implementation of the sleep and wakeup subroutines. (6p)

b. When is the priority used that is given as parameter to sleep? (2p)

c. How is the *wait channel* given as parameter to sleep selected in a typical case? (2p)

Problem 2 (10p)

a. In the original Unix systems, the shell communicated with an RS-232 serial line terminal. Today, a typical computer do not have a serial terminal, but the shell still works in the same way as on the original systems. Describe the mechanism that makes this possible. Also describe the relationship between the shell and the *xterm* terminal emulator. (4p)

b. There is a number of queues used to store characters for character based input and output. There is one queue to store characters for output but two queues for input. Explain why two queues are needed for input and give a short description of how they are used. (4p)

c. Explain why line discipline based 'line editing' cannot be used if the terminal is running in *raw* mode. (2p)

Problem 3 (10p)

a. The page replacement algorithm in FreeBSD is an approximation of *least actively used*. This algorithm uses a reference counter in every page frame. Describe how this reference counter is initialized and updated. (3p)

b. How is it possible that it can be more than one referenced bit set for a page frame at the same time. (2p)

c. What problem can occur when using a physical-address cache together with a virtual memory system? How is this problem solved in FreeBSD? (5p)

Problem 4 (10p)

a. One use of vnodes is to direct a system call to the correct file system code. Explain in detail how this works. (4p)

b. The file system buffer cache in FreeBSD use logical block numbers to identify the blocks. Give a reason for using logical blocks here rather than physical block numbers. How can this logical block numbers be translated to physical block numbers when bringing the blocks into memory? (2p)

c. In some cases there may exist buffers on the CLEAN list that have never been used. Give an example of how this could happen. (2p)

d. In most cases the buffer cache in FreeBSD use page frames to store the data blocks, but there are some exceptions. Explain why page frames are not always used. (2p)

Problem 5 (10p)

a. What is meant with an association when talking about network communication. How is the implementation done to allow for the correct association to be found. (5p)

b. The TCP protocol want to avoid that packets sent by TCP need to be fragmented at the IP level. Describe the method used in TCP to ensure that no IP level fragmentation is needed. (5p)

Problem 6 (10p)

Assume that a one kilobyte UDP datagram has been sent by some process to a remote location at the Internet. Describe the handling of this packet from the call to the *sosend()* routine till it has been sent by the local Ethernet interface. Assume that the Ethernet destination address can be found in the ARP cache. The description shall include where different headers are added and the calls between the different protocol layers.