TDIU25 Exam

Ahmed Rezine

2015-03-17 14-18
TER1

Admitted material

Dictionary from English to your chosen language.

Jour

Ahmed Rezine (013-281938) visiting after about one hour.

Instructions

- Fill in the exam wrapper and read the instructions on it before you start.
  Read instructions and all assignments carefully and completely before you begin.
- You may answer in either English or Swedish.
- State your interpretation of the question and all assumptions you make.
- Write clearly. **Unreadable text will be ignored.**
- Be precise in your statements. **Prove your point when possible.**
  Ambiguous formulations will lead to reduction of points.
- **Motivate clearly and in depth all statements and reasoning.**
  Explain calculations and solution procedures.
- The exam is 40 points and graded U, 3, 4, 5 (**preliminary** limits: 21p, 30p, 35p).
  Points are given for motivations, explanations, and reasoning.

Definitions

Unless you are more specific, the correcting team will interpret the following terms as follow:

memory Volatile random access memory (DRAM), about 100ns access time.

disk Permanent storage, about 10ms access time.

page A fix size region of virtual memory, possibly on disk.

frame A fix size region of physical memory (DRAM).

block A data block located on disk.

Problem 1 (12p)

Assume an architecture with 32 bits physical and logical addresses. The architecture can be configured to use either 4 KiB or 1 MiB page sizes. Assume each page entry takes up 4 bytes.

1. Suppose we adopt 4KiB pages. How many page table levels are needed? What is the smallest size taken by the page tables for a process? you should assume the process has at least one valid page.
2. Same question as 1) for 1MiB pages.
3. For each of the above designs, explain, at each paging level, how many bits can be used in each page table entry for data other than what is strictly required to derive the physical address? give examples of such meta-data that can be associated with the page table entries. Explain how this can be used to prevent buffer overflow attacks.
4. For each design:
   a) what information is cached in the TLB (translation look-aside buffer)?
   b) Assume 20% of the memory accesses are TLB misses (TLB hits take negligible access time), among the misses 0.1% are actual page faults (with 10ms disk access and transfer time). What is the average execution time for one billion (10^9) operations?
5. Draw a diagram that describes how a TLB can be combined with one of your above designs in order to speed up the translation of a logical address to a physical address.

Problem 2 (4p)

For each of the following items, state and explain (in less than 3 sentences) if it is a representative of a capability-based approach or of a ACL-based approach to protection:

1. There is a list of personally invited guests to the Nobel prize banquet.
2. Access to some buildings in Campus Valla after 17.00 requires using a personal access card with a content that matches the list of students and employees at LiU.
3. Cars have to have visible parking tickets. The parking tickets can be obtained by using cash to pay the parking fee.
4. Individual room keys can be used to open a student dormitory or corridor in Ryd.

Problem 3 (12p)

Assume an idle system. Consider the following workload.

Job	Arrival time	Execution time
J_1	0	5
J_2	6	10
J_3	8	9
J_4	20	3
J_5	21	10

Assume a two level scheduler combining two queues: a Round Robin queue with a 4 time units time quantum and a First In First Out (FCFS) queue. The Round Robin queue has higher priority and all jobs are initially inserted at its end. Jobs in the FCFS queue can run only when the Round

Robin queue is empty. They are otherwise preempted. If a process executes longer than 4 time units, it is preempted and inserted at the end of the FCFS queue. If a process running in one of the queues relinquishes (i.e., gives up) the CPU (e.g., for an I/O operation or for sleeping), it is moved to a waiting queue until it is ready. The ready process is then reinserted at the end of the queue it came from before it relinquished the CPU.

1. Draw a Gantt diagram for processes' execution and queues' contents.
2. What is the waiting time and the turnaround time of each process?
3. Give the CPU utilisation of this workload.
4. At time 4, process J_1 is the only process running on the processor and it could continue for one more unit. Yet, it is preempted and moved from the Round Robin queue to the FCFS queue by the kernel. Describe the involved mechanism that makes this possible.
5. How can a programmer who knows the scheduling policy but cannot modify it ensure its long CPU bound process does not starve?
6. How can you modify the multilevel queue to ensure starvation freedom? explain.

Problem 4 (8 p)

Consider a filesystem on a disk that has physical block sizes of 512 KiB and logical blocks that are concatenations of n physical blocks. Assume indexed allocation with 10 direct pointers to data blocks, one indirect pointer, one double indirect pointer and one triple indirect pointer.

1. Assume block pointers are 2 Bytes long. How large should the size of the logical blocks be in order for the inodes to be able to point to any portion of a 1GiB disk? a 1TiB disk?
2. Discuss advantages and inconveniences of the two logical block sizes.
3. Assume block pointers are 4 Bytes long and $n = 2$. What is the maximum size of a file ?
4. Suppose whole blocks are read and written, that operations on blocks are carried in memory, that only direct pointers are used for indexed allocation, that needed files' inodes are already in main memory and you do not need to account for writing them back to disk, and that free blocks are available with known disk positions in memory (no I/O needed to find them). We would like to add one full block of data between the 4^{th} and the 5^{th} data blocks of a file that takes up 8 logical blocks. How many I/O operations on logical blocks are required for each of: (a) contiguous allocation, (b) indexed allocation, and (c) linked allocation.

Problem 5 (4p)

Recall that replay attacks consist in an attacker repeating valid data (e.g., passwords). One mechanism to resist replay attacks in password authentication is to use one-time passwords. For this, a list of n passwords is prepared and shared between the system and the user in a secure way. Passwords can only be used once.
Instead of sharing a big list of n passwords, a system and a user program can instead share a single "main password" pw. They can change pw after each n authentications (the mechanism for changing passwords in a secure manner is not relevant for this problem).
Describe a mechanism that allows a user program and a system to have the same list of n passwords while sharing only one single "main password" pw. Explain how your mechanism can derive the i^{th} password from the current "main password" pw. (hint: one way hash functions)