Challenge 5

Filip Strömbäck, Ahmed Rezine

March 6, 2024

Instructions

You have to do research on the parts you do not understand. When you solve problems, do not hesitate to:

- State your starting point. What do you know from start? Why is it important?
- State your goal. What do you need to know to reach it?
- State your assumptions. What are you taking for granted or assuming?
- Show step by step how you can go from what you have to what you want.

You will get one credit for each presentation you claim prepared. You may (at random) be selected to present any of the solutions you prepared. When you are selected for presentation:

- For each step in your solution, explain what you were thinking. How did you come up with this? What gave you the clues?
- Practice so you are ready to present more than less from memory (you can still have a note for reference, but you should not need to look at it much). **Aim for 10 minutes.**
- You are not required to have a correct solution to get credit, but **you are required to solve all parts of a problem, to make a serious attempt and to "believe" in your solution**.
- If you are not prepared you loose all credit for the seminar in question.

As audience you should think of how difficult it is to clearly present a solution. Be humble and supportive. You may put forward constructive criticism of the presented solution. Compare to your solution; I did it another way, what will be the difference? You are of course welcome to take notes.

Problem 1

Your company has a Linux file server where each division has it's own file area below `/home/<division>`. Now the main system administrator (you) want to delegate responsibility over users and files in the sales division to an administrator, Maria, in that division. Specifically, she will need right to change owner (chown) and change permissions (chmod) on any files below `/home/sales`, but not for the rest of the system.

(a) Explain how you can use the protection features (file protection, ACL's, capabilities[1], setuid programs) in Linux to obtain that goal in the best way possible.

(b) Discuss your solution according to principle of least privilege and need-to-know. Use in your discussion a worst case example scenario.

Problem 2

Suppose two processes need to access the same resource, in this case a file. Assume the operating system provides a way for processes to open files with exclusive access. This means that as long as one process has opened a file in exclusive mode no other process may open it. Both processes use this feature.

Further, we assume that processes run on an embedded device where all processes are correctly implemented and no malicious processes exist. When a process in this system tries to open a file that is busy (no matter who opened the file) the operating system will place that process on hold (on a wait queue for the file) until the file is closed.

Process J_1 opens file `A` for exclusive access after 1 unit of CPU-time (from its start), and then keeps it open during 2 units of CPU-time. Process J_4 opens file `A` for exclusive access after 3 units of CPU-time (from its start) and closes it after another 2 units of CPU-time.

Job	Priority	Arrival time	Execution time	Open A	Close A
J_1	1	6	4	+1	+2
J_2	2	6	3	-	-
J_3	3	5	6	-	-
J_4	4	1	7	+3	+2
J_5	5	3	2	-	-

Table 1: Job arrivals

(a) With regard to the above information, show a Gantt chart for the processes in Table 1 running under preemptive priority scheduling. Assume higher priority numbers correspond to lower priorities (so that J1 has higher priority than J5).

(b) What is meant by turnaround time? Show this for process J_1.

(c) Due to its high priority we would expect process J_1 to finish execution after 4 time units, or in worst case after 6 time units if it must wait for exclusive access to the file for the full duration J_4 use the file. What happen in your schedule?

(d) How can you change the scheduling implementation to fulfill the expectation in previous question?

[1] `man capabilities`

Problem 3

Assume preemptive Round Robin scheduling with a time quantum of 10 units and priority aware. The scheduling algorithm uses a priority queue as a ready queue to organize the processes that are waiting for the CPU. (A priority queue behaves like a FIFO queue for processes with the same priority). Each process is assigned a number to reflect a relative priority. The higher the number, the higer the priority. For instance, process P1 has higher priority than process P5.

A process may be preempted by a higher priority process (even if it did not use all of its time quantum). A preempted process (by a process with a higer priority or at the end of its time quantum) is inserted in the ready queue.

Process	Priority	Burst	Arrival
P_1	40	20	0
P_2	30	25	25
P_3	30	25	30
P_4	35	15	60
P_5	5	10	100
P_6	10	10	105

Table 2: Job data

(a) Use a Gantt chart to show the scheduling of the processes

(b) Give the waiting and turnaround times of P3

(c) Give the CPU utilization in the first 120 time units.

Problem 4

Assume a round robin scheduler with time quantum t ms. For the sake of discussion, assume two processes run "forever" and are the only processes to run on the CPU. One of the processes is I/O bound and one is CPU bound. The CPU bound process is always ready to execute. The I/O bound process uses the CPU to process data, but issues a new I/O request after executing on the CPU for 1 ms. Assume each context switch takes 0.2 ms (not included in the time quantum).

(a) Suppose t is larger than both 1 ms and the time it takes the I/O request to be served. Give the CPU utilization as a function of t.

(b) Can you improve on the CPU utilization by changing the value of t ? Do you see any drawbacks?

Problem 5

Assume a magnetic hard disk with cylinders numbered from 0 to 3999. The head was positioned at cylinder 100 and is now positioned at cylinder 110. The disk controller is to schedule the following sequence of requests: 75, 1471, 719, 1789, 1024, 1619, 1056, 1851, 120.

(a) Give the total distance in cylinders to serve all requests for each of the disk scheduling algorithms: First Come First Served (FCFS), Shortest Seek First (SSTF) and Elevator (SCAN).

(b) Explain advantages/drawbacks (if any) of FCFS compared to SSTF.

Problem 6

A new hacker, whose code name is `Catapault`, has an idea to attack your system. He plans to get hold of the free space manager in the operating system and randomly change some information therein.

(a) What kind of serious problems might occur due to the above-mentioned attack scenario?

(b) Which free space management scheme (bitmap or linked) do you think is more vulnerable to such attack? Provide appropriate justifications.

(c) Assume that the OS uses the bitmap scheme for free space management and the attacker `Catapault` is capable of changing only a small portion of the bitmap. Can you think of an efficient mechanism to detect such attacks and/or recover from the same?

Problem 7

Assume that your friend has designed a data structure library, which includes the implementation of `stack` data structure. In particular, the stack data structure keeps track of the last position in the stack and implements routines to insert any element into the stack (`push`) and delete an element from the stack (`pop`). The pseudocode of this two routines appear as follows:

```
push(x) {
   insert "x" into the last position
}

pop() {
   fetch the value from last position and denote it "x"
   remove the last value from the stack and free it's memory
   return "x"
}
```

As you know the running process user thread may be interrupted at any time due to some hardware such as the timer. In this case the operating system will execute the interrupt handler. It may in turn switch to (schedule) another kernel thread. That kernel thread may in turn resume execution of it's (another) user thread in the same process (1-1 mapping).

If both user threads use operations on the same stack, the execution may become interleaved. For simplicity, assume an interrupt may only occur between two lines in the pseudo-code above, and assume interrupts and thread switches may occur between every line. Then solve the following problems.

(a) Let one user thread execute push and let the other execute pop. List all possible orders in which the operations internal to push and pop may take place. If some sequence will produce unexpected result for either push or pop or both, explain why.

(b) Let both user threads execute pop. List all possible orders in which the operations internal to pop may take place. If some sequence will produce unexpected result for either thread's pop or both, explain why.

Hint: Write each line of pseudo-code on its own post-it note. In how many different sequences can you place the notes? Which of the sequences can occur? (For example, fetch in one thread can not occur after remove in the same thread, the order of execution within one thread can only be the order written in the pseudo-code.) Which sequences will cause inconsistent result? (Compare to the result of calling both functions in a single thread.)

Problem 8

Assume a system with 12-bits virtual addresses. Physical addresses are 16-bits wide. Assume one-level paging with 256-bytes pages. We will consider two processes `P1` and `P2` with page tables Table 1 and Table 2 respectively. The frame numbers are given in hexadecimal, for instance, `0x10` (in hexadecimal) corresponds to 16 (in decimal). In the tables, the value 1 for the valid bit means the corresponding page entry is in memory and valid. Bits other than the frame number and the valid bit are not relevant for the problem.

1. For each one of the following virtual addresses state whether the following virtual addresses are in main memory. If they are, state their equivalent physical address in hexadecimal. All values are given in hexadecimal:
   (a) Virtual address `0x113` for process P1
   (b) Virtual address `0x113` for process P2
   (c) Virtual address `0xCF0` for process P1
   (d) Virtual address `0xCF0` for process P2
2. Do processes P1 and P2 share some frame(s)? If yes, give the virtual addresses of a byte shared by the two processes.

Frame number	Valid bit	Other bits
0x6B	1	XXXXXXX
0x17	1	XXXXXXX
0x60	1	XXXXXXX
0x5B	1	XXXXXXX
0x6A	1	XXXXXXX
0x79	1	XXXXXXX
0x00	0	XXXXXXX
0x6B	0	XXXXXXX
0x97	0	XXXXXXX
0x24	0	XXXXXXX
0x31	0	XXXXXXX
0xA4	0	XXXXXXX
0x72	0	XXXXXXX
0x56	1	XXXXXXX
0x88	0	XXXXXXX
0x1E	0	XXXXXXX

Table 1. Page table of `P1`

Frame number	Valid bit	Other bits
0x56	1	XXXXXXX
0x43	1	XXXXXXX
0x07	0	XXXXXXX
0x1B	1	XXXXXXX
0x44	1	XXXXXXX
0x5B	1	XXXXXXX
0x3C	0	XXXXXXX
0x2A	1	XXXXXXX
0xB6	0	XXXXXXX
0x34	0	XXXXXXX
0x7A	1	XXXXXXX
0x21	0	XXXXXXX
0x4D	1	XXXXXXX
0x2F	1	XXXXXXX
0x09	0	XXXXXXX
0x4C	1	XXXXXXX

Table 2. Page table of `P2`