Chapter 1 exercises

1 How does the distinction between kernel mode and user mode function as a rudimentary form of protection (security) system?

2 Which of the following instructions should be privileged?

a. Set value of timer.

b. Read the clock.

c. Clear memory.

d. Issue a trap instruction.

e. Turn off interrupts.

f. Modify entries in device-status table.

g. Switch from user to kernel mode.

h. Access I/O device.

3 Some CPU s provide for more than two modes of operation. What are two possible uses of these multiple modes?

4 Timers could be used to compute the current time. Provide a short description of how this could be accomplished.

5 Give two reasons why caches are useful. What problems do they solve? What problems do they cause? If a cache can be made as large as the device for which it is caching (for instance, a cache as large as a disk), why not make it that large and eliminate the device?

5 In a multiprogramming and time-sharing environment, several users share the system simultaneously. This situation can result in various security problems.

a. What are two such problems?

b. Can we ensure the same degree of security in a time-shared machine as in a dedicated machine? Explain your answer.

7 What is the purpose of interrupts? How does an interrupt differ from a trap? Can traps be generated intentionally by a user program? If so, for what purpose?

8 Some computer systems do not provide a privileged mode of operation in hardware. Is it possible to construct a secure operating system for these computer systems? Give arguments both that it is and that it is not possible.

9 What is the purpose of system calls?

10 What system calls have to be executed by a command interpreter or shell in order to start a new process?

11 What is the purpose of system programs?

12 What is the main advantage of the layered approach to system design? What are the disadvantages of the layered approach?

13 Describe three general methods for passing parameters to the operating system.

14 What is the main advantage of the microkernel approach to system design? How do user programs and system services interact in a microkernel architecture? What are the disadvantages of using the microkernel approach?