CHAPTER 17

Security

Practice Exercises

17.1 What protection problems may arise if a shared stack is used for parameter passing?

Answer:
The contents of the stack could be compromised by any other processes sharing the stack.

17.2 Consider a computing environment where a unique number is associated with each process and each object in the system. Suppose that we allow a process with number n to access an object with number m only if $n > m$. What type of protection structure do we have?

Answer:
A hierarchical structure.

17.3 Consider a computing environment where a process is given the privilege of accessing an object only n times. Suggest a scheme for implementing this policy.

Answer:
Add an integer counter with the capability.

17.4 If all the access rights to an object are deleted, the object can no longer be accessed. At this point, the object should also be deleted, and the space it occupies should be returned to the system. Suggest an efficient implementation of this scheme.

Answer:
Reference counts.

17.5 Why is it difficult to protect a system in which users are allowed to do their own I/O?

Answer:
In earlier chapters, we identified a distinction between kernel and user mode whereby kernel mode is used for carrying out privileged operations such as I/O. One reason why I/O must be performed in kernel

mode is that I/O requires accessing the hardware, and proper access to the hardware is necessary for system integrity. If we allow users to perform their own I/O, we cannot guarantee system integrity.

17.6 Capability lists are usually kept within the address space of the user. How does the system ensure that the user cannot modify the contents of the list?

Answer:

A capability list is considered a "protected object" and is accessed only indirectly by the user. The operating system ensures that the user cannot access the capability list directly.