Protection

Practice Exercises

14.1 What are the main differences between capability lists and access lists?
Answer: An access list is a list for each object consisting of the domains with a nonempty set of access rights for that object. A capability list is a list of objects and the operations allowed on those objects for each domain.

14.2 A Burroughs B7000/B6000 MCP file can be tagged as sensitive data. When such a file is deleted, its storage area is overwritten by some random bits. For what purpose would such a scheme be useful?
Answer: This would be useful as an extra security measure so that the old content of memory cannot be accessed, either intentionally or by accident, by another program. This is especially useful for any highly classified information.

14.3 In a ring-protection system, level 0 has the greatest access to objects, and level n (greater than zero) has fewer access rights. The access rights of a program at a particular level in the ring structure are considered as a set of capabilities. What is the relationship between the capabilities of a domain at level j and a domain at level i to an object (for $j > i$)?
Answer: D_j is a subset of D_i.

14.4 The RC 4000 system (and other systems) have defined a tree of processes (called a process tree) such that all the descendants of a process are given resources (objects) and access rights by their ancestors only. Thus, a descendant can never have the ability to do anything that its ancestors cannot do. The root of the tree is the operating system, which has the ability to do anything. Assume the set of access rights was represented by an access matrix, A. $A(x,y)$ defines the access rights of process x to

object y. If x is a descendant of z, what is the relationship between $A(x,y)$ and $A(z,y)$ for an arbitrary object y?
Answer: $A(x,y)$ is a subset of $A(z,y)$.

14.5 What protection problems may arise if a shared stack is used for parameter passing?
Answer: The contents of the stack could be compromised by other process(es) sharing the stack.

14.6 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: Hierarchical structure.

14.7 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.

14.8 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.

14.9 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 where 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.

14.10 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 the user cannot access the capability list directly.