17.12 The access-control matrix can be used to determine whether a process can switch from, say, domain A to domain B and enjoy the access privileges of domain B. Is this approach equivalent to including the access privileges of domain B in those of domain A?

17.13 Consider a computer system in which computer games can be played by students only between 10 P.M. and 6 A.M., by faculty members between 5 P.M. and 8 A.M., and by the computer center staff at all times. Suggest a scheme for implementing this policy efficiently.

17.14 What hardware features does a computer system need for efficient capability manipulation? Can these features be used for memory protection?

17.15 Discuss the strengths and weaknesses of implementing an access matrix using access lists that are associated with objects.

17.16 Discuss the strengths and weaknesses of implementing an access matrix using capabilities that are associated with domains.

17.17 Explain why a capability-based system provides greater flexibility than a ring-protection scheme in enforcing protection policies.

17.18 What is the need-to-know principle? Why is it important for a protection system to adhere to this principle?

17.19 Discuss which of the following systems allow module designers to enforce the need-to-know principle.

   a. Ring-protection scheme
   b. JVM’s stack-inspection scheme

17.20 Describe how the Java protection model would be compromised if a Java program were allowed to directly alter the annotations of its stack frame.
17.21 How are the access-matrix facility and the role-based access-control facility similar? How do they differ?

17.22 How does the principle of least privilege aid in the creation of protection systems?

17.23 How can systems that implement the principle of least privilege still have protection failures that lead to security violations?