103011227 周延儒 CS18

1. What are the main differences between capability lists and access lists?

   An access list is a list for each **object** consisting of the domains **with a nonempty set of access rights for that object.**
   On the other hand, a capability list is a list of objects and the operations allowed on **those objects for each domain.**

2. 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.

   We could add an **integer counter** with the capability to implement this policy.

3. 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?

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

4. 14.3 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.

   We could set up a **dynamic protection structure** that changes the set of **resources available with respect to the time** allotted to the 3 categories of users. As time changes, so does the **domain of users** eligible to play the computer games. When the time comes that a user's eligibility is over, a **revocation process must occur.** **Revocation** could be **immediate, selective** (since the computer staff may access it at any hour later), **total**, and **temporary** (since rights to access will be given back later in the day).

5. 14.8 Discuss the need for rights amplification in Hydra. How does this practice

compare with the cross-ring calls in a ring-protection scheme?

Rights amplification is required to **deal with cross-domain calls** where code in the **calling domain does not have the access privileges to perform** certain operations on an object but the called procedure has an expanded set of access privileges on the same object.
Typically, when an object is created by a module, if the module wants to export the object to other modules **without granting the other modules privileges** to modify the object, it could export the object with **those kinds of access privileges disabled.** When the object is passed back to the module that created it in order to perform some mutations on it, the rights associated with the object need to be expanded.
A more **coarse-grained approach to rights amplification** is employed in **Multics**. When a cross-ring call occurs, a set of checks are made to ensure that the calling code has sufficient rights to invoke the target code. Assuming that the checks are satisfied, the target code is invoked and the **ring number associated with the process is modified to be ring number associated with the target code**, thereby expanding the access rights associated with the process.