

Operating Systems exercises – Week 11 – Selected answers.

Tutorial questions

1. Alice can read and write to the file X, can read the file Y, and can execute the file Z. Bob can read X, can read and write to Y, and cannot access Z.

a) Write a set of Access Control Lists for this situation. Which list is associated with which file?

b) Write a capability list of capabilities for this situation. With what is each list associated?

Answer:

a) ACL for

X = (Alice, read/write), (Bob, read)

Y = (Alice, read), (Bob, read/write)

Z = (Alice, execute), (Bob, -----)

b) Capability list for

Alice = (X, read/write), (Y, read), (Z, execute)

Bob = (X, read), (Y, read/write), (Z, -----)

2. In UNIX based systems, when a process uses an `open()` system call to open a file for use, UNIX will check if the process owner (actually effective user id.) has the correct permissions to access the file in the mode requested. If the OS is happy then it creates a file structure for the file which contains the access mode permissions that are granted to the given process and returns to process a file descriptor (an index into table of pointers to file structures). On subsequent calls to perform some operation on the file, the process will access the file using the file descriptor to identify the relevant file structure to the OS, the OS only checks that the permissions contained in the file structure matches the operation requested, it does not check again process ids ownership or usage rights on the particular file.

UNIX uses Access Control Lists. However, does UNIX include capabilities as well as Access Control Lists? [Hint – consider the file descriptor.]

Answer:

If a file is opened and its protection mode is changed, while the file is in use, so as to exclude access by the opener, the process can still access the file using the file descriptor. Thus the file descriptor behaves like a token or authority to access a file in a particular mode much in the same way a capability does. However, it differs from a proper capability based system in that the creation of the capability is temporary and is controlled by reference to an Access Control List.

3. Suppose a user wishes to edit the file XYZZY in a capability based system. How can the user be sure that the editor cannot access any other file? Could this be done in an ACL-based system? If so, how? If not, why not?

Answer:

In a capability based system a capability only contains rights for access by some subject for some particular object. Thus the editor will be granted a capability on the users behalf for the file XYZZY. However, possession of that capability does not permit the editor to access any other object, even if the user who owns the editor process has access permissions on a given object.

This is not true in the ACL based system. Each object will list whether a particular user can access that object in particular modes of access. Thus the editor which is running on the users behalf will be identified as being owned by the user. If the editor makes a request to access some file other than the one it is editing then the OS will only check as to whether the user that owns the editor process has the rights to access that file. If the user does then it will grant access.

4. Revoking an individuals access rights to a particular file is easy when an Access Control List is used. How hard is it to revoke a user's access to a particular set of files, but not all files? Compare this with revocation using capabilities.

Answer:

The system must search through all the files accessing each files' ACL and deleting the entry in the ACL for the particular user. This is time-consuming requiring accesses to many different OS structures (the ACLs) and searching each one for the relevant entries for deletion. In comparison the capability list is associated with the user and one access to that OS structure and one search through it is sufficient for the OS to delete all relevant entries in the capability list.

5. Consider revoking access to a particular file from a group of users, but not all users. How might capabilities fare in comparison to ACLs in this situation?

Answer:

In an ACL based system this is easy as it requires a single access to the ACL to delete the relevant set of entries. However, using capabilities means that the OS must search through the capability lists of each user in the group of users. This is time-consuming again requiring accesses to many different OS structures and searching each one for the relevant entry for deletion.

Virtual Machine exercises.

No answers required as all programs given in exercise sheet in full.