structures to make them available. The naming scheme varies by operating system. Once mounted, the files within the volume are available for use. File systems may be unmounted to disable access or for maintenance.

File sharing depends on the semantics provided by the system. Files may have multiple readers, multiple writers, or limits on sharing. Distributed file systems allow client hosts to mount volumes or directories from servers, as long as they can access each other across a network. Remote file systems present challenges in reliability, performance, and security. Distributed information systems maintain user, host, and access information so that clients and servers can share state information to manage use and access.

Since files are the main information-storage mechanism in most computer systems, file protection is needed. Access to files can be controlled separately for each type of access—read, write, execute, append, delete, list directory, and so on. File protection can be provided by access lists, passwords, or other techniques.

**Practice Exercises**

11.1 Some systems automatically delete all user files when a user logs off or a job terminates, unless the user explicitly requests that they be kept. Other systems keep all files unless the user explicitly deletes them. Discuss the relative merits of each approach.

11.2 Why do some systems keep track of the type of a file, while others leave it to the user and others simply do not implement multiple file types? Which system is “better”?

11.3 Similarly, some systems support many types of structures for a file’s data, while others simply support a stream of bytes. What are the advantages and disadvantages of each approach?

11.4 Could you simulate a multilevel directory structure with a single-level directory structure in which arbitrarily long names can be used? If your answer is yes, explain how you can do so, and contrast this scheme with the multilevel directory scheme. If your answer is no, explain what prevents your simulation’s success. How would your answer change if file names were limited to seven characters?

11.5 Explain the purpose of the **open**() and **close**() operations.

11.6 In some systems, a subdirectory can be read and written by an authorized user, just as ordinary files can be.

   a. Describe the protection problems that could arise.

   b. Suggest a scheme for dealing with each of these protection problems.

11.7 Consider a system that supports 5,000 users. Suppose that you want to allow 4,990 of these users to be able to access one file.

   a. How would you specify this protection scheme in UNIX?
b. Can you suggest another protection scheme that can be used more effectively for this purpose than the scheme provided by UNIX?

11.8 Researchers have suggested that, instead of having an access list associated with each file (specifying which users can access the file, and how), we should have a user control list associated with each user (specifying which files a user can access, and how). Discuss the relative merits of these two schemes.

Exercises

11.9 Consider a file system in which a file can be deleted and its disk space reclaimed while links to that file still exist. What problems may occur if a new file is created in the same storage area or with the same absolute path name? How can these problems be avoided?

11.10 The open-file table is used to maintain information about files that are currently open. Should the operating system maintain a separate table for each user or maintain just one table that contains references to files that are currently being accessed by all users? If the same file is being accessed by two different programs or users, should there be separate entries in the open-file table? Explain.

11.11 What are the advantages and disadvantages of providing mandatory locks instead of advisory locks whose use is left to users' discretion?

11.12 Provide examples of applications that typically access files according to the following methods:
   - Sequential
   - Random

11.13 Some systems automatically open a file when it is referenced for the first time and close the file when the job terminates. Discuss the advantages and disadvantages of this scheme compared with the more traditional one, where the user has to open and close the file explicitly.

11.14 If the operating system knew that a certain application was going to access file data in a sequential manner, how could it exploit this information to improve performance?

11.15 Give an example of an application that could benefit from operating-system support for random access to indexed files.

11.16 Discuss the advantages and disadvantages of supporting links to files that cross mount points (that is, the file link refers to a file that is stored in a different volume).

11.17 Some systems provide file sharing by maintaining a single copy of a file. Other systems maintain several copies, one for each of the users sharing the file. Discuss the relative merits of each approach.
11.18 Discuss the advantages and disadvantages of associating with remote file systems (stored on file servers) a set of failure semantics different from that associated with local file systems.

11.19 What are the implications of supporting UNIX consistency semantics for shared access to files stored on remote file systems?

Bibliographical Notes

Database systems and their file structures are described in full in [Silberschatz et al. (2010)].

A multilevel directory structure was first implemented on the MULTICS system ([Organick (1972)]). Most operating systems now implement multilevel directory structures. These include Linux ([Love (2010)]), Mac OS X ([Singh (2007)]), Solaris ([McDougall and Mauro (2007)]), and all versions of Windows ([Russinovich and Solomon (2005)]).


DNS was first proposed by [Su (1982)] and has gone through several revisions since. LDAP, also known as X.509, is a derivative subset of the X.500 distributed directory protocol. It was defined by [Yeong et al. (1995)] and has been implemented on many operating systems.

Bibliography


