A Possible Implementation of Access Control Lists

Alan Kam
June 14, 1974

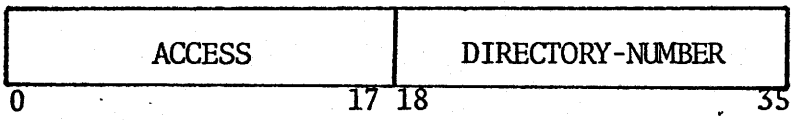
Technical Memo

TM.74-22

In this section, we describe a possible implementation of access control lists (ACLs) for Tenex. We will discuss briefly the composition, maintenance and usage of access control lists for each of two TENEX objects and show how these lists will augment the protection currently provided by Tenex for those objects.

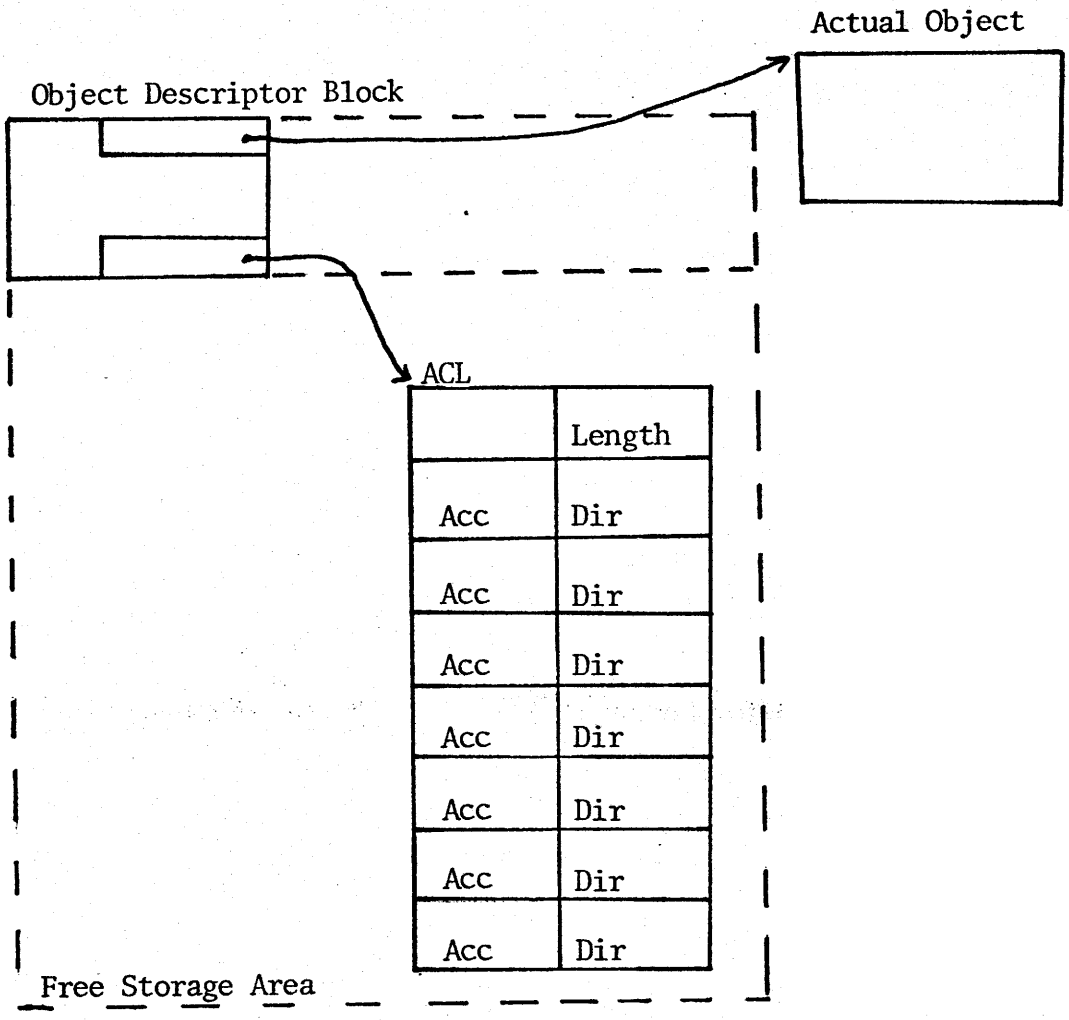
Composition of an Access Control List

An access control list will be a varying length contiguous array of one-word entries. Each one-word entry will have two 18-bit fields, one called the "directory-number" field and one the "access" field as pictured below:



Access control lists will be "associated" with either of two TENEX object types, and each object instance will have its own ACL. The "association" will be by means of a pointer in the object's "descriptor block" to the ACL.

Of course, since the ACL will be of varying length (it can expand and contract), one word of the list (say, the first word) will contain the current length of the ACL. Thus we picture an object and its associated ACL as follows:



The two object types with which we will associate ACL's are TENEX directories and TENEX files. The ACL's themselves will reside in the "free storage area" of each descriptor block and thus will be manipulated by existing storage management routines for allocating, deallocating, compacting and expanding free storage space.

As mentioned, each one-word entry in the ACL will have two fields. The directory-number field will contain a value which is presumed to be the number of a directory currently known to the system. (By this we only mean that it will not contain a value greater than the maximum directory number assigned to date. It may, however, contain the number of a defunct directory.) Directory numbers must be unique and have a

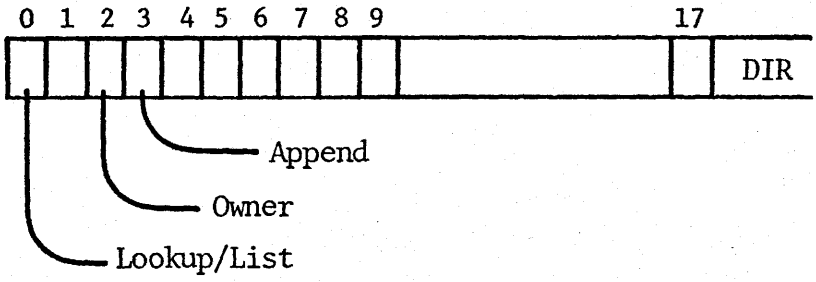
value less than, say, 2^{17} (~100,000). The uniqueness must be forever or until TENEX disappears (whichever comes first).

The access field will contain a bit coding of the accesses which are to be granted to a process whose "name" is the value found in the directory-number field of the ACL entry. In general, a process is "named" by one (or both) of two values: the login directory number, or the connected directory number. The memo on "Management and Protection of Directory Numbers" deals with these considerations in more detail.

The interpretation of accesses encoded within the access field will differ with each of the two object types, of course, since the two types may be manipulated in different ways. We describe the accesses applicable to each in turn.

For a Directory:

The access field will look as follows:



Three accesses are defined to date (see Table 1), and these correspond exactly to the existing accesses allowed for directories now supported under Tenex. A bit setting of 1 means the access is to be permitted.

Bit ∅: Lookup/List access: a user with Lookup/list access is allowed to read the name of any file within the directory.

Bit 2: Owner access: a user who is granted owner access to a directory is allowed to perform operations such as :

a. insert protection information in the TENEX protection word for the directory

b. change certain FDB fields of all Files within the directory

c. insert account information in the directory block

d. connect to the directory (without password verification)

In addition, we propose to allow a user with owner access to

e. manipulate entries on the directory's ACL

f. manipulate entries on the ACL of every file in the directory

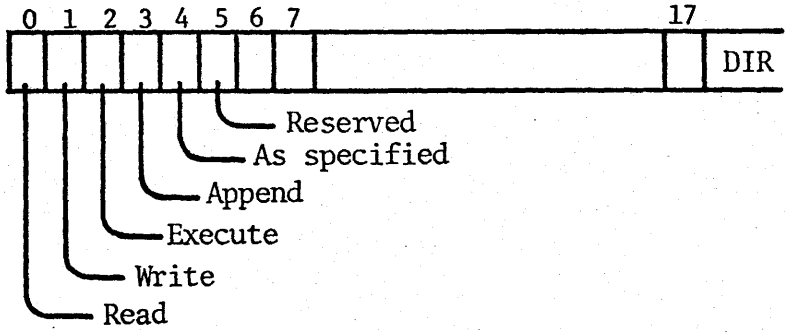
Bit 3: Append access: a user with append access may create files within this directory.

In addition to these 3 accesses, it is practical to think of adding others. For example:

Bit 4: Archive access: a user with archive access may read all the files in the directory without updating the "referenced" date on any of them.

For a File:

The access field will look as follows:



Five accesses are defined to date (see Table 1), and these correspond exactly to the existing accesses allowed for files now supported under Tenex. These accesses are familiar to Tenex users and are hence not discussed here. As in the case of directories, however, it is possible to define new accesses to files such as:

Bit 6: Delete access: a user with delete access may delete the file

Bit 7: List access: a user with list access may retrieve the name of the file

Bit 8: Archive access: a user with archive access may read the file without updating the last referenced date

Bit 9: Owner access: a user with owner access may manipulate the access control list of the file.

Access Bit No.	Directory	File
0. READF	Lookup/List	Read
1. WRTF		Write Delete file Rename file Change FDB
2. XCTF	Owner Protection insertion Change FDB Account insertion Connect directory	Execute
3. RNDF	Append New name New extension New version	Append-write
4. ASPF		As specified in page table
5.		

Table 1

Access bits and their use

(Current implementation)

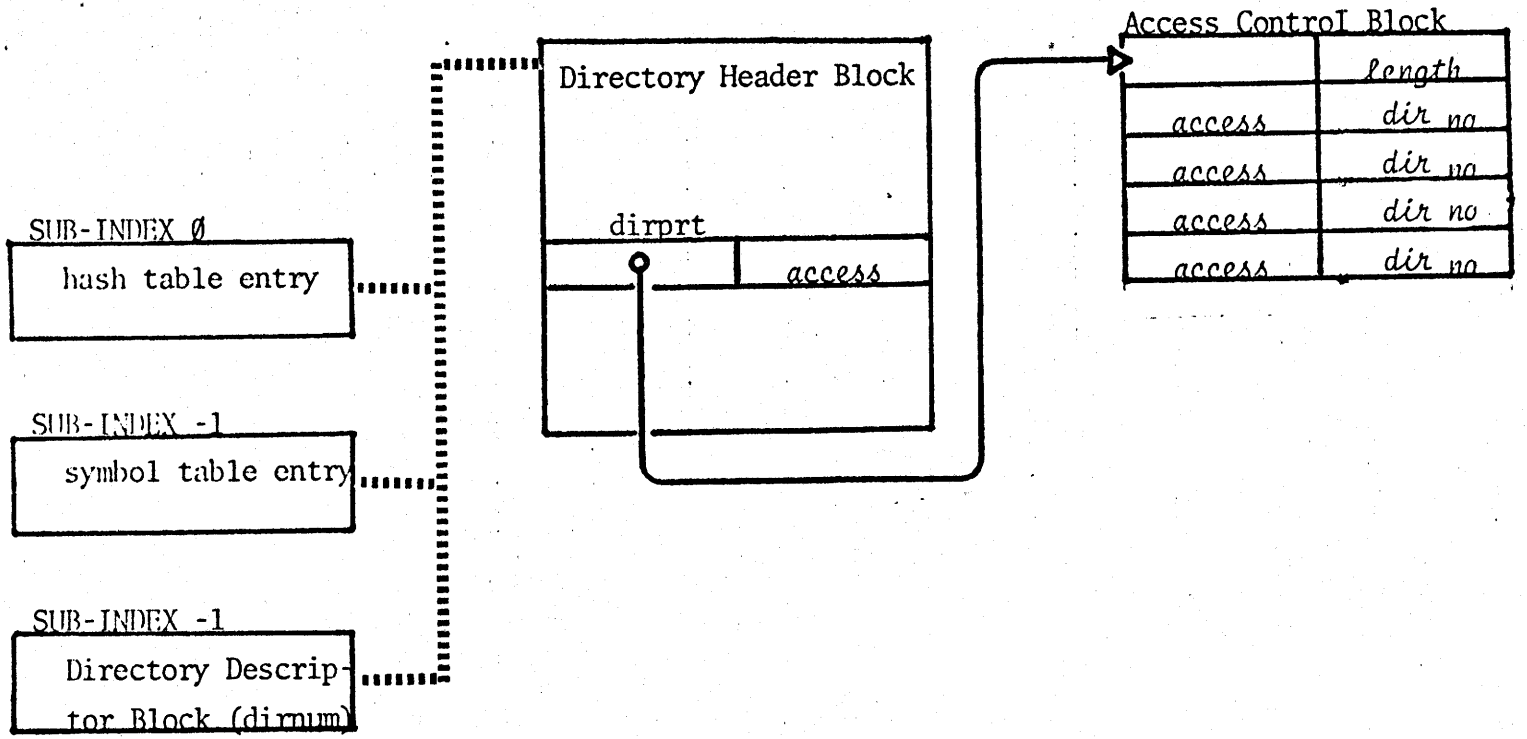
Maintenance of Access Control Lists

We review some ACL particulars:

For directories the left half of DIRPRT, and for files the left half of FDBPRT, might be used to hold the address of the access control list [or, block] in the associated free storage area. If the address is Ø, then a block has not yet been allocated. The right halves of DIRPRT and FDBPRT will continue to hold the Tenex access matrix of 18 bits (3 relationships by 6 types of accesses). The following two figures depict the structure.

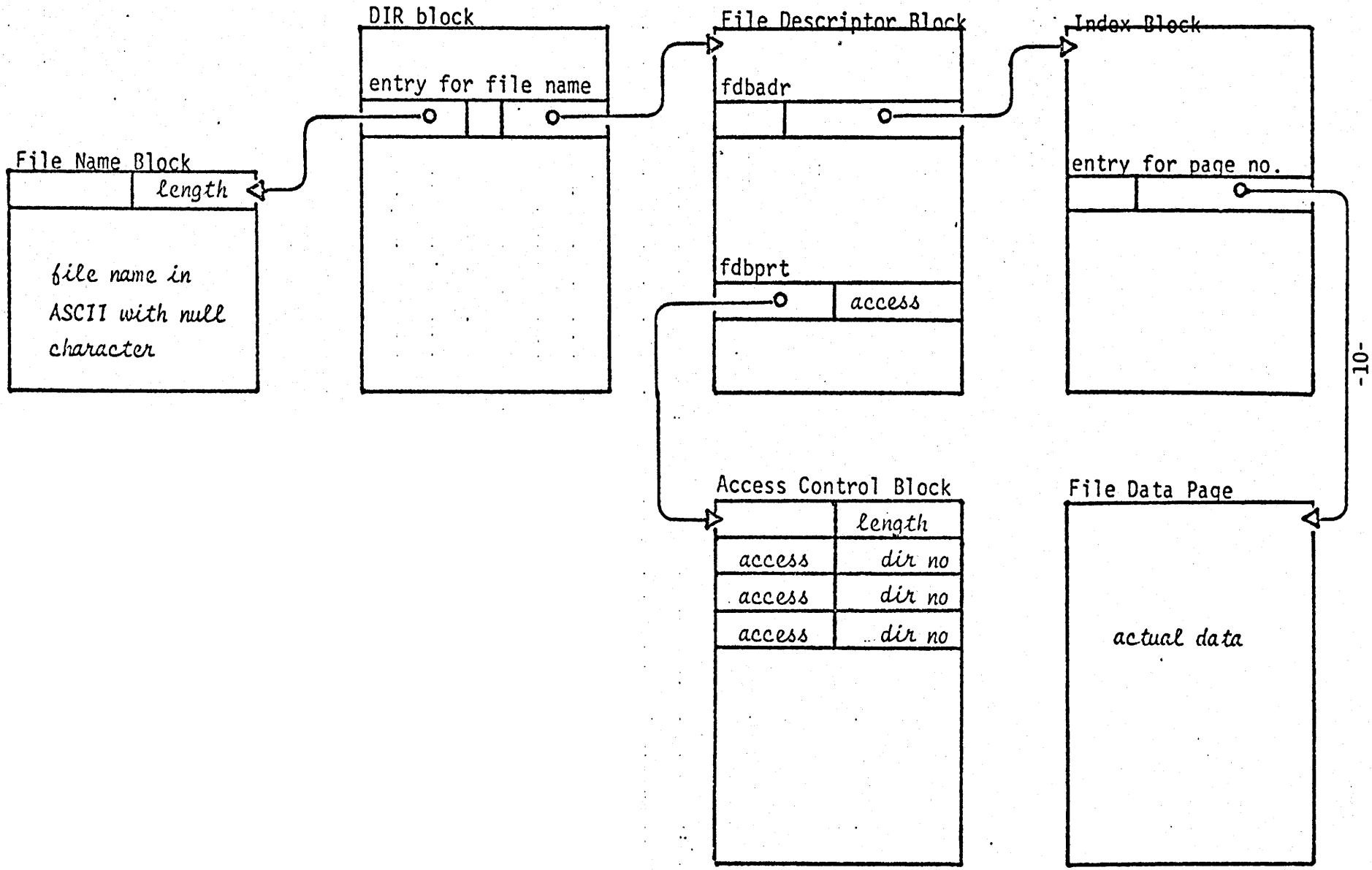
TENEX file system

Access Control Block for directories



TENEX file system

Access Control Block for files



Two new TENEX JSYSes will be required to allow manipulation of access control lists: one to SET an entry and one to inspect an entry. A simple bit setting allows you to address either a directory ACL or a file ACL with the same JSYS.

Descriptions of the two JSYSes (RDACL, SETACL) follow.

RDACL Read Access Control List ENTRY

Accepts in 1: a) 400000_8,, JFN . . to Read a file ACL entry

b) 0,, DIRECTORY-NUMBER to Read a directory ACL entry.

2: entry request of the following forms:

a) Ø,,N to read ACL entry for JFN- or directory-number N.

b) $4ØØØØØ_8$,,N to read the N^{th} entry in the access control list.

An entry in the access control list for the file or directory whose directory-number is specified in AC 1 is returned to the user. If the entry request specifies reading by directory number, then the access control list will be searched for the occurence of the directory number. If the specified directory number does not exist on the list, the JSYS will give a failure return. If the entry request specifies reading of a specified entry number, then the specified entry is returned in the user's AC1. If the specified entry number is less than Ø or greater than the number of entries, the JSYS will fail. (Note that zeroed entries may exist within the ACL.)

Returns

+1: Error, the specified entry was not found. The error codes applicable are the following:

RDACX1: Directory number not found

RDACX9: Invalid entry number, out of range

+2: The entry is placed in the user's AC1 and has the following format:

LH: Access bits

RH: Directory number

SETACL — Set the access control list

Accepts in 1: access control entry
LH: Access bits
RH: Directory number

2: a) 400000_8,, JFN — To set an entry on the ACL of a file

b) Ø,, Directory-number — To set an entry on the ACL of a directory

Search the access control list of the object whose JFN or directory-number is specified in AC2 for the entry containing the given directory number (of AC1).

(1) If there is no access control entry for the given directory number, then add a new one containing the given access and directory number.

(2) If there is a corresponding entry, change its access part to the specified value.

(3) If the new access value is Ø, then the corresponding entry is deleted from the list.

(4) If all entries are deleted from the list, the access control list is deleted.

Returns +1: Failure, job does not have owner access to the file or directory.

+2: Success.

Usage of Access Control Lists

The Access Control Lists will be used as follows by DIRCHK and ACCCHK.

(1) Access to the object will be computed by the current TENEX scheme. If the resulting access is insufficient, the appropriate ACL will be scanned.

(2) The ACL will be scanned as follows:

(a) The directory number in an ACL entry will be compared with the caller's "connected" and "login" directory numbers (actually LH and RH of FRKDIR. See the memo on "Management and Protection of Directory Numbers".) If either of these numbers matches the directory number in the ACL element, the access bits in the element will be merged in with the ones computed in step (1). If the result is still insufficient, the scan continues.

(b) Scan terminates as soon as the computed access is sufficient.

(c) Routines fail only if entire list is scanned without getting required access.

There are several variations to the implementation of ACL's which can provide for more efficient encoding (by grouping); Also, ACL's provide a convenient way to implement individual access restrictions for each of the 36 administratively-defined TENEX groups. (There are several ways to handle this last problem which range from no to significant modifications of monitor code).