MANAGEMENT AND PROTECTION OF DIRECTORY NUMBERS

Jack Freeman
June 14, 1974

Technical Memo

TM.74-24

Management and Protection of Directory Numbers

The addition of ACLs to the File System is valuable in itself and can be put to good use independently of other changes to Tenex. But our goal is to provide a means by which users may safely run programs which exercise capabilities not directly exercisable by the users themselves, and File System ACLs are just one of the primitives needed to achieve this.

We elsewhere describe how, when a program is put into a fork in preparation for execution, it may "bring with it" various capabilities. Among these capabilities will often be a directory number which the program will use to get access to files through the new ACL mechanism. How is this "bringing along" of a directory number and subsequent use of it to be implemented?

At first sight it seems that what we want is to "connect" the fork we put the program in to the directory specified by the directory number. This seems to fit in fairly neatly with the current scheme of things. However, some potential problems come to mind. It seems very probable that programs will need to use both the directory number they bring with them and the directory number of the user who invokes them. Furthermore, cases are sure to arise where a program which has brought with it a directory will want to start up another program which brings with it yet another directory. This second program may well need to use its own directory, the directory of the program that started it, and the directory of the user.

Considerations like these suggest the need for a reasonably general mechanism for managing the directory numbers associated with the forks of a job. We suggest that these numbers be explicitly recognized as a new type of job-local object and that they be managed analogously to JFNs and protected by the same general sort of protection mechanism as is proposed for JFNs and forks. The following sketch of an implementation is meant to make this suggestion clearer.

New Data Structures

A. Job Directory Table

In the JSB of each job will be a table called, say, DIRTAB, whose function is to provide storage for directory numbers accessible to forks of the job. Assuming a limit of 18 forks per job, each entry in this table will be a single word with the following format.

Ø 17	18 35
protection data	directory number

The protection data field is an encoded access control list for the directory number and is interpreted as follows:

Fork i can <u>use</u> the directory number if and only if bit i is set in the protection data field.

How directory numbers are "used" will be explained below.

DIRTAB will be big enough to hold a reasonable number of directory numbers, say 8.

B. Fork Directories Word

In the PSB of each fork will be a single word called, say, FRKDIR. This word has the format:

∅ 17	18 35
index into DIRTAB	index into DIRTAB

and thus specifies two directory numbers for use by the fork.

New JSYSes

A. Reading and Setting FRKDIR

Read FRKDIR (RFDIR) will return the FRKDIR word for a specified fork. It will fail if the given fork handle is illegal or if the fork issuing the JSYS has insufficient access to the specified fork.

Set FRKDIR (SFDIR) will set the FRKDIR bit for a specified fork to a specified value. The left and right half-words of the specified value apply to the left and right half-words of FRKDIR respectively. The legal values for each half-word are given below, with their meanings.

777777:	leave corresponding half-word of FRKDIR unchanged.
Ø:	clear corresponding half-word of FRKDIR.
$1 \le N \le 8$:	set corresponding half-word of FRKDIR to N.

Fails if

1. fork handle illegal.
2. calling fork has insufficient access to specified fork.
3. calling fork has insufficient access to specified ($1 \le N \le 8$) DIRTAB entry.
4. specified fork has insufficient access to specified ($1 \le N \le 8$) DIRTAB entry. (The reason for this restriction is explained below.)

Note that RFDIR could be incorporated into SFDIR at little cost. All that would be required is that SFDIR return the new value of FRKDIR and that, in the special case where the value specified by the caller is 777777 in both halves, SFDIR checks to see if the calling fork has "read FRKDIR" access to the target fork instead of checking for "write FRKDIR" access.

B. Reading and Setting DIRTAB

Read DIRTAB (RDIRTB?) returns the contents of a specified entry in DIRTAB. Its argument is an integer, N, between 1 and 8. Fails if DIRTAB[N] is in use (≠ Ø) and calling fork's bit is not set in LH of DIRTAB[N].

Set DIRTAB (SDIRTB?) can only be used to modify the left half of a DIRTAB entry, i.e., only to modify the protection data. Its arguments are an integer, N, between 1 and 8 and a word with 777777 in its right half and the desired new protection data in its left half. It fails unless the calling fork's bit is set in LH of DIRTAB[N]. Further checking is done on the value of the new protection data presented. The rules are:

1. It is legal to leave any bit unchanged.
2. It is legal to turn on bit i only if the calling fork has "add capability" access to fork i.
3. It is legal to turn off bit i only if the calling fork has "delete capability" access to fork i.

Management of DIRTAB and FRKDIR

A. Creation of a Job (LOGIN)

When a new job is created DIRTAB[1] and DIRTAB[2] are initialized to the Login Directory.

B. Creation of a Fork (CFORK)

The bit corresponding to the new fork is set in the protection data fields of DIRTAB[1] and DIRTAB[2]. The new fork's FRKDIR word is set to (2,,1).

C. Connection to a Directory (CNDIR)

The directory number field of DIRTAB[2] is set to the newly connected directory's number.

D. Fork Destruction

When a fork is deleted, it's bit is turned off in all DIRTAB entries. If this results in the LH of the entry becoming Ø, the RH is also set to Ø, and the DIRTAB entry becomes "free".

E. Explicit Release of a DIRTAB Entry

The new JSYS, SDIRTB, can be used to turn off bits in the LH of DIRTAB entries. When bit i is turned off in DIRTAB[N], two management functions are performed.

1. Any occurence of N in the FRKDIR word of fork i is changed to Ø.
2. If the LH of DIRTAB[N] becomes Ø as a result of turning off bit i, the RH of DIRTAB[N] is also set to Ø.

Note that the rules above, together with restriction (4) on the use of SFDIR, are intended to insure that there can never be a fork whose FRKDIR word points to a "free" entry in DIRTAB.

Use of FRKDIR and DIRTAB

A. DIRCHK and ACCCHK

The directory numbers pointed to by the two halves of FRKDIR will be used (in place of JOBDIR and JOBDNO) in determining whether to use the "self" field in the directory and file protection words. They will also be used in the extensions to these functions which search file and directory access control lists.

B. Default Directory

The directory number pointed to by the LH of FRKDIR will be used as the "default directory" by GTJFN et. al.

C. Directories of "Protected Programs"

When a program which brings with it its own directory number(s) is gotten into a fork (by a new JSYS), DIRTAB gets used as follows. For each directory number to be brought in, a free entry in DIRTAB is found. The directory number is put in its RH, and the bit corresponding to the new fork is set in its LH. The indices of the DIRTAB indices thus assigned may be put directly into the fork's FRKDIR word or "passed" to the fork as arguments.