	title  MODEL I FILE SYSTEM		prefix/class-number.revisio
DCC			MIFS/W-10
checked		authors	approval date revision date
checked		Rainer Schulz	classification Working Paper
approved .		diamit whint	distribution Company Private 9

## ABSTRACT and CONTENTS

This document describes in general the functions that are available in the basic system to manipulate files and other objects kept in MIB's. A detailed description of the format of the basic system calls (including parameters, returned values and error conditions) is described in another document.

bcc

p/c-n.r MIFS/W-10 1

Every user in the Model-I system has one multiindex block (2K page) which mainly serves as a combination file directory and index block for the user.

In order to fully understand the operation and protection of the file system a description of the contents of the MIB follows:

## MIB Contents (Figure 1)

- 1) Lock on MIB
- System version for MIB
- User Profile

Header

- 4) Owner Access Lock List
- 5) Public, Friend Access to MIB
- 6) A list of users who may access the MIB's files independently of the public or access key mechanism (Friend Table).
- 7) Object Table, specifying which files, access keys, processes and other objects are kept track of in this MIB.

bcc

p/c-n.r page MIFS/W-10 2

The object table can point to the following types of objects:

- 1) small file
- 2) large file
- 3) access key
- 4) process
- 5) data (arbitrary data of any size from Ø to 250 words maximum)

The SIB blocks for these objects are shown in Fig. 3

Every object has access fields for

the public

the owner of the MIB (see below)

friends

and may also have an access control list, each entry of which consists of an access field and the value of an access key. An access field has 4 bits, which allow

R reading

W writing

X execution

O ownership

of the objects. For objects other than files, some of the bits may have slightly peculiar meanings.

When an attempt is made by sub-process S to access an object, the access to be allowed is determined by the first of the following access fields which is not null (TAK(S)) is the temporary access key of the sub-process).

bcc

p/c-n.r page.
MIFS/W-10 3

- if TAK(S)  $\neq \emptyset$  and TAK(S) appears in entry i on the access list of the object, then the access field of entry i
- if TAK(S) = the owner of the MIB, then the owner access field of the object
- if TAK(S) is on the friend list for the MIB, then the friend access field of the object  $\land$  the access field of the friend list entry.
- if TAK(S) = the user number for the process and the
   account number for the process is on the friend
   list for the MIB, then the friend access field of
   the object \( \) the access field of the friend list
   entry for the account.

the public access field of the object

The "owner access" lock list gives the ability to create objects and to set the access to objects in the MIB to anyone who presents a key which matches one of the locks. This list is of fixed size and contains 3 entries. These entries are just like locks on objects in SIB's. In principle, being on the list gives complete control, since by setting the access of an object one can gain control of it.

Links (pseudonyms) are implemented with objects of type data.

bcc

P/c-n.r page MIFS/W-lo 4

## Operations on MIB

The following operations are possible on MIB's or their contents. All calls on the basic system involving MIB's take a user number as argument. This user number serves to identify the MIB. All names given to objects in MIB's must be unique. Trying to create an object whose name already appears in the MIB for another object is an error.

- Read entry E: returns the contents of the entry,
   E is a number which indexes into the object table, or a name of an object. Requires some access to object.
- 2) Set name of entry n to m: n is the name of the object. The name gets set to the new name m. Requires owner access to object.
- 3) Set public, friend, owner access of entry n to value v: n is the name of the object. The access to the object gets set to v. Requires owner access to object or MIB. The entry addressed may be the MIB.
- 4) Set lock and access for lock on entry n: n is the name of the object. The lock is a 40bit value. Requires owner access to object or MIB.
- 5) Delete entry n: n is the name of the object. Requires owner access to object. If object is a process or a file, it cannot be deleted if there are still data pointers in the object.

bcc

**P/c-n.r Page** MIFS/W-10 5

- owner access to MIB. Creates a new entry in MIB and sets up entry like "set name of entry n to m" operation, except that n is not given.

  The index of the object in the MIB is the value of this function. The type of the entry must be given in the call.
- 7) Make new friend f with access v: requires owner access to MIB. Takes user or account number and access bits and makes an entry in friend table. If the entry already exists, access bits get changed to new value.
- 8) Delete friend f: requires owner access to MIB.
  Removes specified user or account from friend table.
- 9) Read friend table: read access to MIB required.
- 10) Read profile. The contents of the profile is not yet defined.
- 11) Set profile: requires special capability includes initialization of MIB.
- 12) Set value of object n to v: requires owner access to object. Works only for objects of type "data." WFI has to be set if UNO(S) is different from user number of process.
- 13) Copy access key n to TAK of sub-process: requires read access to access key.

p/c-n.r page MIFS/W-10 6

- 14) Copy object n of MIB m to a new object in MIB k:
  requires read access to object n and owner
  access to MIB k. Does not work for files or
  processes. In case of access keys, if the R
  bit is set, and W is not, the access key gets
  frozen in MIB k (cannot be copied any more).
  If the W bit is set, the key does not get frozen. If the frozen bit is on in MIB m (bit
  Ø of the access key value), then the access
  key cannot be copied.
- 15) Set no drum charge flag for file: requires owner access to file and special status.
- 16) Read available space in MIB n. Returns as value the number of words not used in SIB area of MIB. Requires some access to MIB.
- 17) Set lock and access for lock in owner access
  lock list: this operation requires owner access to MIB. If lock already exists, access
  is set to new value. If the new value is zero,
  lock gets removed.
- 18) Set reentrant flag for file: requires owner access to file.

A caller is considered to be the actual owner of an MIB only if his UNO equals the access key which is the first entry on the owner access lock list.

bcc

P/c-n.r page MIFS/W-10 7

Owner access to an object, like read and write access, is determined through the friend list, public or owner access, or through the access key mechanism.

The first entry on the owner access lock list of every MIB is initialized with an access key whose value is the user number of the person owning the MIB. This key cannot be removed from the list.

Access keys have unique values (i.e., every time an access key gets created, a unique value is supplied by the system).

There are no operations for copying access keys attached to sub-processes into MIB's.

## Manipulation of Files

In order to access data in a file, the file has to be "opened." All open files have an entry in the OFT table (Fig. 4) which is kept in the context block. OFT has 16 entries. The following operations on OFT entries are possible (sub-process (S) is assumed to make the call):

- 1) open file f: works only if OF bit is set in the status word. PR in the status word has to be set if the file has X access in its access field. f consists of:
  - a) user number of file's owner
  - b) name of file. Fails if the access field of the file is null.

bcc

p/c-n.r page MIFS/W-10 8

The status bit WFI is ANDed with the W bit in the access field of the file if the user number of the MIB that contained the file is different from the user number of the process. The result of the AND is put in the W bit of OFT. The AL and CL fields are set to NAME(S). The OFT entry is created and its index in the OFT is returned

- 2) Set access lock (AL) for entry n to m: if  $CL(N) \land KEY(S) \neq \emptyset$  then AL(N) can get set to any bits.

  Else if  $AL(N) \land KEY(S) \neq \emptyset$ , then  $AL(N) \blacktriangleleft M \land KEY(S) \land AL(N) \land KEY(S)$
- 3) Set control lock (CL) for entry n to m: legal if  ${\rm CL}(N) \wedge {\rm KEY}(S) \neq \emptyset. \quad {\rm File \ gets \ closed \ if \ CL} = \emptyset$  after this operation.
- 4) Set word length for entry n: if AL(N) ∧ KEY(S) ≠ Ø, and W = 1, then the low order 11 bits of word length in the file length word of the SIB are set.
- 5) Read entry n: returns contents of entry n.
- 6) Create new page n of file m: if  $AL(M) \cap KEY(S) \neq \emptyset$  and if W = 1, then a new page gets created in position n. The call fails if the page already exists.

bcc

p/c-n.r page MIFS/W-10 9

- 7) Delete page n of file m: if AL(M)  $\land$  KEY(S)  $\neq \emptyset$ , and if W = 1, then page n gets deleted.
- 8) Give next page after page m of file n: returns the next data page of file n if AL(N) ∧ KEY(S) ≠ Ø. Returns (-1) if no more pages.
- 9) Put RN of page n of file m into PMT entry k with RO access in PMT set to the value of w in OFT: if AL(M) ∧ KEY(S) ≠ Ø, CL(PMT(k)) ∧ KEY(S) ≠ Ø, and PMT(k) is empty, the real name of the file page is put into PMT. The FP bit is set.

It should be noted here that a file can get closed by setting the CL of the OFT entry to zero. This can happen by calling on the file system to change the CL of the OFT entry, or by deleting a sub-process. When a sub-process gets deleted its name gets removed from all locks and keys. If this operation results in the CL of a file getting set to zero, the OFT entry is removed.