TO: Distribution

FROM: T.H. VanVleck

DATE: June 26,1974

SUBJECT: Adding Support for Secure Removable Disk Packs to Multics

The Multics supervisor currently does not permit the dynamic addition or removal of disk volumes to the system configuration while the system is in operation. All disk storage in the configuration is "owned" by the Storage System, which requires that all volumes be on-line when the system is started and that all volumes remain on-line.

There are several reasons for wishing to modify the current arrangement. Applications which require a large data base that is not used all the time now have the choice of on-line disk storage or of tape storage. On-line disk is expensive, partly due to the need for channels, disk controllers and disk drives which must all be configured permanently, and it is difficult for an installation to increase its configuration on short notice. But data stored on tape can be accessed only linearly, and cannot be used by more than one process at a time.

Allowing data stored on disk to be mounted and dismounted also provides some conveniences which may be quite important. A disk which is not mounted cannot have its contents damaged by system crashes or hardware failures, so removable packs will be useful for providing cheap backup of large groups of segments. The transport of data from one installation to another might also be facilitated if packs could be carried, although this suggestion ignores the fragility of disk packs.

GOALS

We can suggest several reasonable goals for the implementation of removable disk pack support.

First, data stored on a removable pack should be accessed in exactly the same way as data on a permanent part of the system's disk storage, once the pack is mounted. If this goal is not met, we will need different sets of programs for the manipulation of each class of data, and applications will need complicated ad hoc code if they are to be able to handle either type of data.

Second, the addition of removable disk pack support should not introduce any compromise of the system's ability to protect information. Access control for data on removable packs should be as effective as the standard access control mechanisms for on-line data, and on-line data's security should not be compromised by the implementation of removable data.

Third, the effort required to implement should not be incommensurate with the value of the facility.

APPROACHES

There are four possible ways to approach implementation of removable pack support.

No Removable Disk

This is the current situation. No new security problems are involved, and effort to implement is zero.

Disk as Per-User Device

This approach sounds simple, but turns out to be complicated. Basically, certain disk packs are not controlled by the storage system at all, but are instead handled just the way tape storage is now. Protection of the information can only be provided by insuring that only one process at a time may access the pack, and by using the same strategy as that proposed for tape. User programs which access these data do so through iox_ attachments through a new DIM for disks. Packs may be in any format that the MPC can handle. The problems arise with the assumption that the file system and the user of a private disk can be insulated from each other. This change is a fundamental departure from the original design of the disk DIM, and will probably require a complete redesign of the disk DIM. Although such an ability would be desirable in order to support on-line disk T & D, its implementation may be very difficult. Effective use of this facility in an Access Isolation environment would probably require an implementation of a "disk daemon" like the proposed "tape daemon", defeating the whole reason for adding removable disk.

Worse yet, this solution does not provide the degree of transparency for user code which is desired. Data on removable disks cannot be shared by more than one process, or accessed by regular segment addressing.

Removable Subtrees

Another option might be the following: constrain the page-assignment algorithm of the current file system to place all pages inferior to a special directory on the removable pack, and only such pages. Then entire subtrees could be removed by dismounting the pack. When such a pack was mounted, the FSDCT would be updated with the pack's free storage map, and all file maps on the pack would be relocated to contain proper addresses.

This solution provides transparency, in that when data is on-line it works just like permanent storage. Multiple users may share data on such a pack, for reading and writing.

However, some problems are introduced. Since directories can be dismounted and remounted, and since the file system tends to crash if a directory is damaged, every directory on the pack must be completely "salvaged" - that is, checked for validity and rebuilt if necessary - every time the pack is mounted. All filemap addresses must be checked to make sure that they are legal, and relocated to correspond to the new (temporary) area number assigned at mount time.

The security problems this approach generates are non-trivial. First, care must be taken to examine every directory entry in order to check that unauthorized modification while the pack is not connected to Multics has not introduced a security hole, such as a gate into ring 1. The level and category of each segment and directory must also be validated. If a pack is carried from one site to another, all ACL entries must be re-interpreted.

Although the implementation of this proposal sounds easy, there are many concealed problems. For example, the dismounting of a pack obviously cannot be done until all pages in core and on the bulk store have been flushed back to the disk. This in turn requires some sort of lock to prevent re-activation of segments and directories while the flush is being done. The same sort of lock or test will be needed to detect references to segments when they are not on-line. Furthermore, care must be taken to insure that the system forbids one dismountable subtree as an inferior of another, to prevent the attempt to deactivate the parent of an active segment.

The impact of this mechanism on the quota machinery may be complicated. Special action must be taken to prevent moving quota generated on a removable pack back into the permanent hierarchy.

Finally, the subtree restriction will require user programs to organize their data in a peculiar fashion. User data which can be dismounted will have to reside in a strange section of the hierarchy, and users will have to organize their trees and access permissions in a strange way.

Dismountable Storage Volumes

The fourth solution to supporting dismountable packs is a combination of the other approaches. For this solution, we require that directories remain on-line, and that only segments can be dismounted. This allows the system to maintain complete control over the security-related attributes of a segment, and to eliminate problems which would arise if a portion of the path of a segment were dismounted. The information contained in directories now will be physically split into two groups: the tree structure, names, ACL's, and so on will remain in the directory, while the disk addresses in the file maps, and other attributes of the physical storage such as current length and date and time modified will be collected into a special storage area on the pack itself called the Volume Table of Contents (VTOC).

This approach fits in well with the already-planned storage system modifications intended to enhance system function and reliability described in MTB-017 and MTB-055. By making each pack self-describing, so that addresses on a given pack are never written on any other pack, the new Storage System limits the damage which may result from a re-used address, and allows packs to be mounted without requiring relocation (though some validity checking must be performed). This approach is the one we are now giving serious consideration.

IMPACT ON SECURITY CONTROLS

Since directories cannot be dismounted in the proposed implementation, there is no significant difference in system security for the data items kept in the directory branch between the old and new storage systems. The ACL, access class, etc., have not been moved at all, and the file map has simply been moved to a different disk record. The same argument can be applied to all other data on "permanent" volumes that cannot be dismounted. Furthermore, as long as dismounted volumes remain in the computer room, they are as safe as the backup tapes and so no new exposure is introduced.

When volumes are removed from the central facility, some more analysis is needed. We require that the pack labeling mechanism be sufficiently careful to prevent accidental interchange of volumes and that the system be able to check a pack for validity before using the volume's contents in order to prevent crashes due to invalid pointers, VTOC entries, file maps, etc.

If users of more than access authorization can create segments (of different access classes) on a given removable volume, or if users of different authorizations may request the mounting and dismounting of the same volume, there will be several possibilities for communication between different access authorizations. Therefore, dismountable volumes will be restricted to contain information with one access class only. The access class will be recorded in the on-line volume registration data, and mount requests will be rejected if the access authorization is incorrect.

The access class will also be written in the pack label, for the convenience of BOS. The pack initialization utility will always completely clear a disk pack when rewriting the pack label.

Permanently-mounted volumes need not be subjected to the single-level rule; it is assumed that quotas are set so that no user may run the system disks out of space. (I suppose we could have a system-parameter option which enforced the rule for permanent volumes, if any installation was enthusiastic about buying that much disk.)

The volume-dismount message for disks should print out the access class of the disk to remind the operator to take appropriate procedures when returning it to storage.

The use of disk packs for transfer of data between different installations requires that new branches be constructed at the receiving site and made to point to the VTOC entries on the carried pack. This operation could obviously lead to nasty security breaches, and crashes, if users were permitted to perform it without appropriate checks. The temporary solution proposed is either to forbid this operation entirely or to make it a highly-privileged function requiring SSA intervention.