

To: Distribution

From: T. H. Van Vleck, W. S. Silver

Date: February 24, 1976

Subject: Interim version of mount and demount for disk volumes

Until the full scheme for user mounting and demounting of hierarchy volumes (storage system disk logical volumes) can be implemented, an interim scheme must be used. The full plan, described in MTB-229, will involve modifications to RCP to know about two new resource types, code in ring 1 to allocate disk drives to user requests analogous to that now used for tape, several new user commands, and new volume registration commands and operator commands. The full details of the eventual mechanism's user, librarian, and operator interfaces will be described in several forthcoming MTB's.

The interim mechanism is much simpler. From the operational point of view it appears to be a slight extension of the current operator commands for system startup. Since RCP is bypassed completely, the initial facility will lack any mechanism for allowing a user to await the mounting of a hierarchy volume; if a requested volume is not mounted, his call will fail immediately. When the user requests the mounting of a volume via telephone, or such mounting is scheduled, the operator must select a free disk drive and mount the pack, and then must type a command to the system indicating that the volume is mounted. He then tells the user, "try it now," and the user issues the mount command.

The "virtual mount" described in MTB-229 will be implemented. This change insures that correct access control discipline is obeyed by preventing a process from accessing segments on a volume unless the volume is public or the process has the volume id in its KST. The user's access to the hierarchy volume will be checked in ring 1 when the mount request is issued, and the virtual mount will be done from ring 1 if the user has access according to the volume registration data. Registration data will be checked for all volumes except the RPV; during a cold boot, the operator will register other volumes before using them. The registration data will not be the full data base which will eventually be implemented for RCP. One segment per hierarchy volume will be maintained in a directory under >system_control_1 (the root hierarchy volume's registration segment will reside in the root directory). These segments will contain enough information to allow ring 1 to check that the

Multics Project internal working documentation. Not to be reproduced or distributed outside the Multics Project.

hierarchy volume is correctly mounted: that is, public/private switch, AIM information, and a list of the physical volumes and their unique ID's. Every private volume must also have an Access Control Segment (ACS), which will be linked to from the system directory. This segment is a (possibly zero length) segment in a location specified by the volume owner. The ACL of the ACS is interpreted as specifying access to the hierarchy volume contents.

STEPS IN MOUNTING

To get a hierarchy volume virtually mounted to his process, a user first contacts operations and asks that the volume be mounted. This request may be conveyed via telephone or send_message, or it may be implicit in a schedule established by the system administrators.

Operator Preparation

When the operator decides to mount a volume as a result of such a request, he may choose a free disk drive if one is available, or he may use the following command to force a mounted hierarchy volume to be demounted:

```
demount_force <hvname>
```

This command will cause one or more physical volumes in use by the storage system to be shut down in an orderly fashion. All active segments on the volumes will be deactivated, and the label, VIOC, and free map for the volume will be updated. Users who attempt to use segments on a volume which has been demounted will encounter a seg_fault_error condition with the message "Volume not mounted." The supervisor will also print a message of the form

```
DEMOUNTED DSK7_04
```

giving the disk drive name.

When the operator has sufficient free disk drives to mount the requested hierarchy volume, he performs the physical mounting operation for each pack. Special interrupts generated by the disk units becoming ready will be ignored. The operator then types the following command for each physical volume he has mounted:

```
add_volume <pvname> <drive-name>
Example: add_volume pack32 dsk7_04
```

This command directs the initializer to call ring 1, where the following steps are taken:

- a) The combination <drive-name> is looked up in the disk_table, and that table entry is checked to make sure it is available for storage system use and currently free.
- b) The disk_table is also searched to insure that <pvname> is not mounted on some other drive.
- c) The registration information for <pvname> is located. Each per-hierarchy-volume registration segment has additional names of the form "pv.<pvname>" added to it for every physical volume in the hierarchy volume. If the physical volume is not registered it cannot be mounted.
- d) Ring 1 now calls the hardware to read and check the label of the pack on <drive-name>. The physical and hierarchy volume names and ID's and the AIM attributes are checked to make sure that the label matches the registration data. If everything matches the volume is accepted for paging and entered into the PVT.

When the operator has mounted all physical volumes he then issues the following command:

```
mount hvol <hvname>
Example: mount hvol student3
```

This command causes the initializer process to call ring 1 to cause the following steps to be performed:

- e) The registration data for hierarchy volume <hvname> is located.
- f) For each physical volume recorded in the registration of <hvname>, the disk_table is checked to insure that the physical volume is mounted. If the physical volume is recorded in the disk_table as assumed, steps c and d above are performed.
- g) The hierarchy volume <hvname> is entered in the hardware LVT by a call to initializer_gate_\$add_hv. If the registration data says the volume is public, any user process may then use it without further ado.

User_Call

For a private volume, the user must now cause the virtual mounting of the hierarchy volume for his process. To do this, he may invoke the rcp-oriented command

```
mount hvol <hvname>
```

In a later RCP implementation, many other options and subcases of the command will be possible. But for the interim version, the

command simply calls the two entrypoints

```
rcp_$mount ("hvol", ifp, event, "", id, ec)
```

```
rcp_$check_mount (id, ifp, "", ix, ec)
```

in order. These entries will be called with their final standard calling sequence, but the rcp_gate will direct these calls to interim code which does not perform all the actions which the final version will perform. The following steps are taken:

- h) The registration information for <hvname> is located.
- i) The ACS for the hierarchy volume is located and the user's effective access to the hierarchy volume is derived. If the user does not have RW access to the hierarchy volume, an error is returned.
- j) Ring 1 calls the hardcore to check that the hierarchy volume is in the LVT. If the volume is found this call will also enter hierarchy volume ID into the process's KST, unless the volume is a public volume.

The user process may then initiate segments on the hierarchy volume.

When the user has finished with a non-public volume, he may issue the command

```
demount hvol <hvname>
```

It will remove a user's KST item for a hierarchy volume and cause faults to be set in the SDW's of any active segments on the hierarchy volume for the user process. This operation can decrement a counter in the LVT which was counted up by the mount operation, so that the ring-1 programs can type

```
VOLUME STUDENT3 FREE
```

when the count becomes zero.

System Startup

The current temporary mechanism for system startup will be modified as follows:

- a) The DSKA command will be replaced by the add_volume and mount commands described above.
- b) The DSKG command will be eliminated.
- c) The automatic DSKG performed by certain commands such as startup, salv, and reload will be changed to be an automatic

"mount hvol root" command.

- d) If the special volume name "auto" is used in a mount command from the Initializer process, the disk_table will be scanned for volumes which are assumed to be in position but which have not yet been checked. Each hierarchy volume thus found will be mounted as described above. Installations which leave Multics running unattended can therefore place the command "mount hvol auto" in their system_start_up.ec to cause all volumes which were in use at the last crash to be rechecked and reaccepted automatically.
- e) Simple registration commands will be available that can be executed in the cold boot environment to register at least the volumes which are part of the RHV. These commands will be consistent with the commands used by the volume librarian, although the librarian commands may have more options.
- f) The initialize_disk command will be renamed initialize_volume. Only registered volumes can be initialized.
- g) When the system mounts a volume automatically because the disk_table shows that it was mounted at the time the system crashed, a registration file with default attributes will be generated if the volume appears to be unregistered. Thus, if the volume registration data is destroyed in a crash, it is reconstructed from the table of correctly mounted volumes if that data has survived.

The regular system startup procedure will thus differ from that used in 28-0 only by the detail that the system accepts hierarchy volumes other than the RHV from ring 4 rather than ring 1. More typing is required during a cold boot, since the volumes must be registered; and more typing is required after a disk reshuffle, since mount commands as well as add_volume commands must be typed.

DECLARATION OF THE VOLUME REGISTRATION FILES

The following PL/I declaration describes the structure of the interim volume registration segment.

```

dcl 1 volume_registration aligned,
    2 version fixed bin,
    2 hvid bit (36),
    2 hvname char (32),
    2 max_access_class bit (72),
    2 min_access_class bit (72),
    2 volume_owner char (32),
    2 flags,
    3 public bit (1) unal,

```

```

3 pad bit (35) unal,
2 npv fixed bin,
2 pv ( ) refer npv),
3 pvid bit (36),
3 model fixed bin,
3 pvname char (32),
3 location char (32),
3 mfg_serial char (32),
3 date_registered fixed bin (71);

```

All volume registration segments except that for the RHV will reside in the directory >system_control_1>hvol. The RHV's registration segment will reside in the root directory, to insure that it is accessible while the system is coming up, before it has accepted the other volumes of the RHV.

Each volume registration segment will be named hv.<hvname> and will have additional names added to it of the forms hvio.<unique string> and pv.<pvname> and pvid.<unique string> for each physical volume. This is done so that master directory control can associate volume ID's with their volume names, and to insure the uniqueness of volume names.

The ACS for each hierarchy volume is a link in >sci>hvol with the name <hvname>.acs. The link target for all public volumes is a zero-length segment in the same directory, with an ACL of rw for *.*.*.

Similarly, the Master Directory Control File (MOCF) for the hierarchy volume will be pointed to by a link with the name <hvname>.mdcf.

NEW_OPERATOR_COMMANDS

This section gives brief descriptions of the new operator commands available in rings 1 and 4.

add_volume_registration (avr)

This command calls the rco_llb_ gate which is accessible to volume librarians and to the Initializer.

Format: avr pack <pvname> -hvol <hvname> -user <userid>

-hvol <hvname> This control argument is required. When avr is called from ring 1 the only legal <hvname> is "root".

-user <userid> This control argument is required. When avr is called from ring 1 the only legal <userid> is "system".

initialize_volume

This command writes a label and an empty volume map and VTOC onto a disk pack. It consults the volume registration for the volume to obtain the hierarchy volume information and the physical volume unique ID. It also checks the label of the volume and will refuse to label a volume if it appears to have a valid label for a registered volume.

Format: initialize_volume <pvname> <drive-name>

When initialize_volume is called from ring 1 only volumes of the RHV may be initialized.

-special If this control argument is specified, the system will ask the operator for request lines which may specify average segment length and partition definitions. The valid requests are:

part NAME low nrec	Define a partition on the volume at the low end.
part NAME high nrec	Define a partition on the volume at the high end.
avg fff.ff	Declare the average segment size to be ffff.ff records. (The default is 4.1.)
list	List the attributes of the volume.
quit	Exit without doing anything.
end	End of specifications; initialize the volume.

add_volume (addv)

This command is issued to inform the system that a volume is mounted and ready on a specified disk unit.

Format: addv <pvname> <drive-name>

No control arguments are allowed.

mount

This command is issued to inform the system that a hierarchy volume is completely mounted.

Format: mount hvol <hvname>

demount_force (dmf)

This command forces the demounting of a hierarchy volume.

Format: dmf <hvname>

One or more physical volumes will be demounted.

IMPLEMENTATION

This section gives a summary of the programs which must be modified or written for release 4.0.

1. Modifications to RCP.

- a) Fix rcp_device_info_ to accept device names of the form "uskX_01".
- b) Install new version of rcp_init_disk_sharing_ which respects the flag pvt.storage_system.
- c) Fix rcp_disk_ to read the label of IO disks and refuse to work on storage system packs except for privileged mount requests.

2. User commands.

mount
demount

3. User subroutines.

rcp_\$mount
rcp_\$check_mount
rcp_\$demount

4. Operator and Librarian commands. mount

demount_force
add_volume
del_volume
add_volume_registration
del_volume_registration
change_volume_registration
list_volume_registration

5. Operator and Librarian subroutines.

rcp_sys_\$demount_force
rcp_lib_\$set_volume_registration
rcp_lib_\$copy_volume_registration

system_startup_
rcp_vol_data_name
rcp_vol_data_uid
disk_table_
initializer_admin_

6. Hardcore

LVT manager
various checks that volume is mounted or public