Title
NOTES ON CREATING A VAX/VMS CLUSTER

Author
Harvey, E.H.

Publication Date
1988-11-01

LBL-26257 c.2
Preprint

Lawrence Berkeley Laboratory
UNIVERSITY OF CALIFORNIA

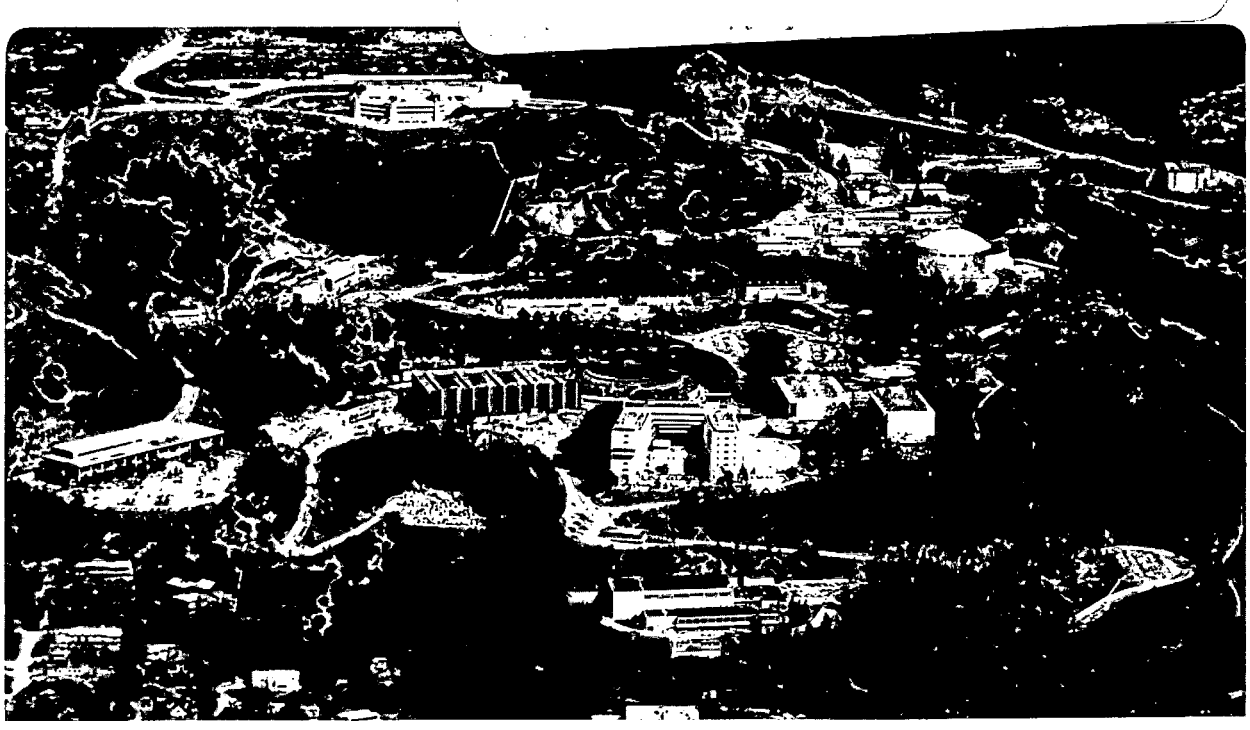
Engineering Division

Submitted to DECUS SIG Newsletter

Notes on Creating a VAX/VMS Cluster

E.H. Harvey, Jr.

November 1988



Prepared for the U.S. Department of Energy under Contract Number DE-AC03-76SF00098.

DISCLAIMER

NOTES ON CREATING A VAX/VMS CLUSTER

Everett H. Harvey, Jr.

Engineering Division
Lawrence Berkeley Laboratory
1 Cyclotron Road
Berkeley, CA 94720

"This work was supported by the Director, Office of Energy Research, Office of High Energy and Nuclear Physics, Division of High Energy Physics, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098."

CONTENTS

This article describes how to convert a non-clustered system disk to a clustered disk structure without re-installing VMS. The notes were accumulated over the months of creating a cluster from three previously independent VAX computers. Some comments about Local Area VAX Clusters (LAVC) are included. As a supplement to the DIGITAL documentation on VAXclusters, we concentrate on problems encountered and solutions found in areas where the DIGITAL documentation was weak.

With VMS version 5 all system disks will have the clustered disk structure. To ease conversion problems you may want to create these structures under VMS version 4. The procedures described here may help you.

These notes are divided into several major sections:

1. Creating a common system disk: We had running VMS systems and did not want to go through a VMS upgrade. Also, we did not want to start with an intial installation of VMS. A new installation would mean integrating modified files - such as systartup and authorization files. Waiting for the next release of VMS was not acceptable, and even more important, we had already formed a cluster with each node having its own system disk. Now the HSC had arrived and a common system disk was possible. Doing an upgrade on one node would remove that node from the cluster (all nodes should be at the same level of VMS). We show how to create a common system disk structure for one of the nodes.

2. Adding nodes to the common system disk: How to get all of those other nodes using the common system disk made in step 1. Trying to make the independent machines run from the same system disk requires planning and lots of work. DIGITAL documentation describes how to add nodes as though you are unpacking a brand new CPU. What are some of the snags in integrating one that has even had different system management? A minimum goal is to arrive at a VMS-homogeneous system so that the next update to VMS can be applied smoothly. This requires that there be only one copy of critical VMS files - such as the authorization files.

3. Problems with served disks: Although we ran into these problems when forming the cluster before having an HSC, this also applies to Local Area VAX Clusters. So we include it here, just before the LAVC section. Also, many sites want to preserve their current non-HSC based disks and have them available cluster-wide.

4. Local Area VAX Clusters: Local Area VAX Clusters became available under VMS 4.6. Users already at VMS 4.6 need to convert to a clustered system disk structure to install the LAVC software.

1 CREATING A COMMON SYSTEM DISK

1.1 REASONS FOR A COMMON SYSTEM DISK

One of the goals of a common system disk to minimize system management effort. If only one copy of key files are maintained, many of these tasks can be reduced. For instance having one SYSUAF.DAT file for the whole cluster may be advantageous. SYSUAF.DAT files may be merged by the CONVERT utility ($ CONVERT/MERGE sysuaf1,sysuaf2,...sysuafn mastersysuaf).

However, it is possible to still have each node use an independent authorization file. Understanding how this would work gives an example of the search list features of the cluster directories. Each node has a boot root directory. A non-clustered system root is usually SYS0. For example on the node booted from SYS1 a SYSUAF.DAT in SYS$SYSDEVICE:[SYS1.SYSEXE] will be used instead of the SYSUAF.DAT in SYS$SYSDEVICE:[V4COMMON.SYSEXE]. Alternative names for the directories are: SYS$SPECIFIC:[SYSEXE] which corresponds to SYS$SYSDEVICE:[SYS1.SYSEXE] ; and SYS$SYSDEVICE:[V4COMMON.SYSEXE] corresponds to SYS$COMMON:[SYSEXE]. A reference to SYS$SYSTEM will mean: look first in SYS$SPECIFIC:[SYSEXE] and if the file is not found there look in SYS$COMMON:[SYSEXE]. That is, a SYSUAF.DAT in a node's specific directory will be used and the common one ignored. SYS$SPECIFIC: refers to the boot root directory.

Thus, you could have a cluster with several nodes using a shared SYSUAF.DAT in SYS$COMMON:[SYSEXE] and other more independent nodes each with a separate SYSUAF.DAT in their respective SYS$SPECIFIC:[SYSEXE] directories.

1.2 CREATING THE COMMON SYSTEM DISK STRUCTURES

If you are not doing an initial installation or upgrade then use the VMSKITBLD.COM procedure to create a common system on a new target disk. If at all reasonable, use an upgrade procedure on older disks for this eliminates the need to move files from your old disk to the target disk. This section describes steps using VMSKITBLD.COM

WARNING: If you have modified ANY of the system files, such as STARTUP.COM, DCLTABLES, HELPLIB, etc., you will get these modified versions on the target disk. As you add new nodes you will need to pay particular attention to differences in these files. If any of these files are modified you may want to start with an original VMS distribution.

You will need a free disk drive as the target drive will be reinitialized. If necessary, format and check the target disk pack

for bad blocks.

Execute the VMSKITBLD.COM file. An example execution follows.

```
$ @sys$update:vmskitbld

VMSKITBLD - Build or Copy VAX/VMS binary distribution kit for
                                                        Release 4.

Operation [BUILD,ADD,COPY,COMMON]? COMMON
Enter mounted SOURCE disk name (DDCU:): $1$dra3:
Enter SOURCE disk top level system directory [default = SYS0]:
Enter TARGET disk name (DDCU:): $2$dua3:
Enter the TARGET disk's label [default = VAXVMSRL4]:
Enter TARGET disk top level system directory [default = SYS0]:
  It will be necessary to initialize the target disk(s).
Is the target disk, $2$DUA3:, ready to be initialized? (Y/N): Y
%DCL-I-ALLOC, HSC000$DUA3: allocated
%MOUNT-I-MOUNTED, VAXVMSRL4    mounted on $2$DUA3: (HSC000)
Create directory entries on the target disk.
Create the SYSGEN files.
%SYSGEN-I-CREATED, $2$DUA3:[SYS0.SYSEXE]SWAPFILE.SYS;1 created
%SYSGEN-I-CREATED, $2$DUA3:[SYS0.SYSEXE]PAGEFILE.SYS;1 created
%SYSGEN-I-CREATED, $2$DUA3:[SYS0.SYSEXE]SYSDUMP.DMP;1 created
Copy the system executive files.
Copy the system library files.
Copy the system message files.
Copy the system manager files.
Copy the system update command files.
Copy the system EXE files.
Copy the system help files.
Write a bootblock.
Copy BLISS require files and STARLET SDL library.
Copy coding examples.
Copy the UETP files.
Kit is complete.
```

1.3 COPY OTHER FILES

Now mount the new target disk and copy over needed files.

The following command file shows an example of files to copy

```
$ SET VERIFY
$! update new CommonSystemDisk with files from source
$! (Adds files not copied by VMSKITBLD.COM
$!  - the assumption is that SYS0 is the root directory on both)
$!
$!!! SET NOVERIFY
$ INQUIRE SRC "What is the SourceDisk"
$ INQUIRE TGT "What is the TargetDisk"
$!
$! Authorization databases,etc (latest version)
$!
$ COPY/LOG 'SRC'[SYS0.SYSEXE]SYSUAF.DAT 'TGT'[V4COMMON.SYSEXE]
$ COPY/LOG 'SRC'[SYS0.SYSEXE]NETUAF.DAT 'TGT'[V4COMMON.SYSEXE]
$ COPY/LOG 'SRC'[SYS0.SYSEXE]RIGHTSLIST.DAT -
                                        'TGT'[V4COMMON.SYSEXE]
$ COPY/LOG 'SRC'[SYS0.SYSEXE]VMSMAIL.DAT 'TGT'[V4COMMON.SYSEXE]
$!
$! and make sure there is at least one sysgen parameters file
$ COPY/LOG 'SRC'[SYS0.SYSEXE]VAXVMSSYS.PAR -
                                        'TGT'[V4COMMON.SYSEXE]
$!
$! Keep going...as errors are expected from here on
$ SET NOon
$!
$! Executable Images and CommandFiles not previously moved
$!
$ BACKUP/VER 'SRC'[SYS0.SYSEXE].EXE; -
               'TGT'[V4COMMON.SYSEXE].;1/OWNER=ORIG
$ BACKUP/VER 'SRC'[SYS0.SYSEXE].COM; -
               'TGT'[V4COMMON.SYSEXE].;1/OWNER=ORIG
$!
$! Get all the specific sysmgr stuff
$!
$ BACKUP/VER 'SRC'[SYS0.SYSMGR...].; -
               'TGT'[SYS0.SYSMGR...]/OWNER=ORIG
$!
$! Get the specific stuff from sysexe
$!
$ BACKUP/VER 'SRC'[SYS0.SYSEXE...].COM; -
               'TGT'[SYS0.SYSEXE...]/OWNER=ORIG
$ BACKUP/VER 'SRC'[SYS0.SYSEXE...].DAT; -
               'TGT'[SYS0.SYSEXE...]/OWNER=ORIG
$ BACKUP/VER 'SRC'[SYS0.SYSEXE...].PAR; -
               'TGT'[SYS0.SYSEXE...]/OWNER=ORIG
$!
$! Get any missing stuff from other sysxxx
$!
$ BACKUP/VER 'SRC'[SYS0.SYSLIB...].; -
```

```
                'TGT'[V4COMMON.SYSLIB...].;1/OWNER=ORIG
$ BACKUP/VER 'SRC'[SYS0.SYSHLP...].; -
                'TGT'[V4COMMON.SYSHLP...].;1/OWNER=ORIG
$ BACKUP/VER 'SRC'[SYS0.SYSERR...].; -
                'TGT'[V4COMMON.SYSERR...].;1/OWNER=ORIG
$ BACKUP/VER 'SRC'[SYS0.SYSMSG...].; -
                'TGT'[V4COMMON.SYSMSG...].;1/OWNER=ORIG
$ BACKUP/VER 'SRC'[SYS0.SYSUPD...].; -
                'TGT'[V4COMMON.SYSUPD...].;1/OWNER=ORIG
$!
$! Misc [sys0.xxxxx] stuff
$!
$ BACKUP/VER 'SRC'[SYS0.SYSCBI...].; -
                'TGT'[V4COMMON.SYSCBI...].;1/OWNER=ORIG
$ BACKUP/VER 'SRC'[SYS0.SYSMAINT...].; -
                'TGT'[V4COMMON.SYSMAINT...].;1/OWNER=ORIG
$ BACKUP/VER 'SRC'[SYS0.SYSTEST...].; -
                'TGT'[V4COMMON.SYSTEST...].;1/OWNER=ORIG
$!
$! Any users with default device SYS$SYSDEVICE
$!                                (not already there)
$!
$ BACKUP/VER 'SRC':[FAL...]         'TGT'[...]/OWNER=ORIG
$ BACKUP/VER 'SRC':[DECNET...]      'TGT'[...]/OWNER=ORIG
$!
$! Done
$ EXIT
```

1.4 CREATE NEW BOOT MEDIA

Now create a new boot console media (make sure you save your old one!) If you were not previously running from an HSC based disk you will need the HSC node number (not to be confused with a decnet node number as this is strictly a hardware number). See the documentation on booting from a common system disk. Basically the idea is to modify CIBOO.CMD by placing the HSC node number in R2, The HSC disk unit number in R3 and the root number (for the first system on the common system disk this is zero) into R5. If you have dual-ported disks (two HSC's hooked to same disk) then you will indicate both HSC hardware node numbers in R2.

for example

```
.
.
.
DEPOSIT R0      20              ! CI PORT DEVICE
DEPOSIT R1      E               ! CI TREE
DEPOSIT R2      00              ! HSC NODE NUMBER
DEPOSIT R3      03              ! DEVICE UNIT NUMBER
DEPOSIT R4      0               ! BOOT BLOCK LBN (NOT USED)
DEPOSIT R5      00004000        ! BOOT FLAGS
  .             |      |
  .             |      Set this to "1" for conversational boot
  .             |       (use 0 for DEFBOO)
                |
                This designates the SYSn root directory
                 (here n = 0 )
```

You should now be ready for booting from the new disk.

1.5 ACTUAL REBOOT

Do the conversational boot first. In our case we already had been running as a cluster so all of the sysgen parameters had been set. If you are initially creating a cluster you now set the SYSGEN parameters before doing the autogen. Reboot again using autogen:

```
$ @SYS$UPDATE:AUTOGEN SETPARAMS REBOOT
```

After booting you will probably find some problems with your startup command files. Be sure to use sys$manager and sys$system instead of sys$sysdevice:[sys0.sysmgr] and sys$sysdevice:[sys0.system] to locate files. Once you have a good common system disk structure you will then be ready to consider adding additional nodes.

2 INTEGRATING ANOTHER NODE INTO THE COMMON SYSTEM DISK

The following is a prescription for adding a running new node to the common system disk. It is assumed that a common system disk is already in use and that a new node is being added to the common system disk. The new node's current system disk is known as the source disk. The common system disk is called the target disk. While each VAX site has different local variations some general outline of the technique is possible. The prescription that follows is divided into a series of steps. It is assumed that you are logged onto the node with a common system disk structure and that the system disk of the new node to be added is accessible.

2.1 PREPARATION

The first step is information gathering: determine that the nodes are running the same version of VMS - an absolute necessity! The following information must be known about the node to be added to the common-system-disk.

Parameter	How to determine it
Node name	SHOW NET
SCS system ID	1024 * node-area + node-number use NCP SHOW EXEC to get area and node number use SYSGEN to SHOW SCSSYSTEMID directly
pagefile size	use SHOW MEMORY to find current pagefile(s) sizes
swapfile size	use SHOW MEMORY to find current swapfile(s) sizes
dumpfile size	use SHOW MEMORY/PHYSICAL to find current total memory size in pages. Add 4 to this number. If you expect to add more memory,add 2048 for each additional megabyte more.
DECNET accounts	Get full authorization file listings for the default DECNET and any other network accounts. Get a NETUAF listing with the authorize SHOW /PROXY

Decide upon a root for this new node. It should be in the range of hexidecimal 1 through D. Check the current system disk top-level directories to find one not in use.

```
DIRECTORY SYS$SYSDEVICE:[0,0]SYS.DIR
```

It is much better to prepare the new source system before proceeding further. Doing as much as possible before the move saves lots of effort after the move. There are many things that can be done. First a brief list is given and then some of the issues are discussed. Things to do are:

1) Clean up SYSUAF.DAT. Pay particular attention to accounts with default device SYS$SYSROOT or SYS$SYSDEVICE. Get a complete listing and look for conflicts with the current common-system authorization file. It is a good time to check security levels too.
2) Clean up NETUAF.DAT. This file should receive the same attention as SYSUAF.DAT.
3) Clean up RIGHTSLIST.DAT. Again the same attention as SYSUAF.DAT. BE SURE TO GET FULL LISTINGS OF ALL OF THESE FILES.
   USe the authorize utility commands
   ```
   SHOW/RIGHTS/USER=        (for an alphabetic list)
   SHOW/RIGHTS/USER=[,]     (for a UIC ordered list)
   ```
4) VMSMAIL.DAT - not much you can do here, some notices about new mail may be lost. There is a MAILUAF.COM file, in the SYSLIB examples directory, which is useful for cleaning up mail files.
5) SYSTARTUP.COM - try to make sure that all the needed files are in the sys$manager directory. These files should refer to sys$manager and particularly not sys0 or some other directory tree used by some other node.
6) Default and other DECNET accounts. Try to make the new node use the same account(s) and directories as the common system.
7) SYLOGIN.COM is often used along with the logical name SYS$SYLOGIN for system-wide login commands. Make sure that there are no conflicts or unsolved problems.
8) Logical names need attention. Get complete listings of logical names from each system. Check for potential conflicts and make any necessary modifications.

2.2 POSSIBLE SECURITY LEAKS TO PLUG

Note that users can access files anywhere they exist. For instance a person having group access rights on one of the nodes can access files owned by any user in the same group, no matter on which node that other user has his account. Security issues apply to the whole cluster.

2.3 SO, YOU HAVE MODIFIED SOME DEC SOFTWARE!

Any local modifications to DCLTABLES or HELPLIB.HLB should be un-done as they will not (generally) be available on the new common-system. However, you may put copies of the modified files into the SYS$SPECIFIC directories, thus making the nodes dissimilar and defer the problem until later. Of course this will become VERY important to resolve BEFORE applying the next software updates. SOFTWARE UPDATES WILL BE DONE TO THE COMMON VERSION. A good general rule is "do not modify DCLTABLES or HELPLIB.HLB".

Use separate help files and the HLP$LIBRARYn logical names (see $ HELP HELP /USERLIBRARY). However, if you have already done modifications to HELPLIB.HLB, you can recover the original .HLP file from the help library. Use the LIBRARY/EXTRACT command.

If you have new verbs defined in DCLTABLES.EXE, try to get the original .CLD commands used to create the verb. If you do not have them you do not have the utilities with VMS to reconstruct them like the extract command available with help libraries. There is a public domain program which purports to do this and this program is described later.

To create new commands use log-on DCL set-commands instead of modifying DCLTABLES.EXE. Layered products from Digital (and possibly others) can cause real problems here. You should be prepared to install products, available on the new node, onto the common-system disk if they are not already there.

Some files, such as Common Data Dictionary can be merged, however, again detailed understanding of the contents is needed. (Much like SYSUAF.DAT). Trying to maintain separate system files can continuously create problems. For instance if the SYSUAF file is changed in structure by a newer version of VMS you need to make sure that each separate SYSUAF file is converted to the new structure.

2.4 ACTUAL INTEGRATION

It is best to make a backup copy of your current console media. This copy will later be modified to boot from the common system disk.

You will want to have a backup of your common system disk too!

Once you have done all of the above preliminary preparations, you are ready to begin with the actual move. In the following use these definitions:

location	actual reference (n = root number)
sys$specific	targetdisk:[sysn.sysexe]

```
sys$system      sourcedisk:[sys0.sysexe]
sys$common      targetdisk:[v4common.]
```

Now you are ready to create the structures on the common system disk for the new node. You will use the command file MAKEROOT.COM.

Next build a new console media to boot from the common system disk.

Note: The makeroot.com command file will provide some information about creating a boot file on your console media. In particular it says to start with a copy of DEFBOO which assumes that the source node is already running off of an HSC based disk. It is better to start with CIGEN and modify that file. I choose to call the modified file CSDBOO.CMD for Common System Disk Boot. Be sure that the correct parameters are set. Also create a copy of CSDBOO.CMD as DEFBOO.CMD except that DEFBOO.CMD does not have a conversational boot flag set. for example:

```
.
.
.
DEPOSIT R0      20              ! CI PORT DEVICE
DEPOSIT R1      E               ! CI TREE
DEPOSIT R2      00              ! HSC NODE NUMBER
DEPOSIT R3      03              ! DEVICE UNIT NUMBER
DEPOSIT R4      0               ! BOOT BLOCK LBN (NOT USED)
DEPOSIT R5      10004000        ! BOOT FLAGS
 .              |       |
 .              |       Set this to "1" for conversational boot
 .              |             or to "0" for default boot
                |
                This designates the SYSn root directory
                 (here n = 1 )
```

From a node already running from the common-system-disk type

```
@SYS$SYSMANAGER:MAKEROOT.COM
```

Provide the answers as to node-name and SCSSYSID.

Choose sizes determined from the source node for the page and swap files. Later these primary page and swap files maybe reduced in size and secondary files be used from another disk for better performance. Using original source node total size prevents any problems due to these file sizes.

Create the SYSDUMP.DMP file in the sys$specific directory.

Copy all NET.DAT from the source sys$system to the target sys$specific

Copy all PARAMS.DAT from sys$system to sys$specific

Copy VAXVMSSYS.PAR from sys$system to sys$specific. Make sure this

copy is used instead of the one put there by makeroot.com.

Delete the sys$specific VMSIMAGES.DAT. This file SHOULD BE the same as the one in sys$common:[sysmgr] so it is not needed. If it is different be careful. Check it if you are not sure.

Copy all files from the source root (sourcedisk:[sys0...]) to the target root (targetdisk:[sysn...]). If you have plenty of disk space you can move all the files over and then delete the files out of [sysn...] that are already in [v4common...]. Some command files given later help with this approach. Otherwise copy files that do not already exist in either the target root or sys$common:[...].

Assure that the Common Data Dictionary (if any) is in sys$specific. Later you may want to move it to sys$common or merge it with other Common Data Dictionaries into sys$common.

Use BACKUP/IGNORE=INTERLOCK to assure that SYSUAF.DAT, NETUAF.DAT, RIGHTSLIST.DAT, and VMSMAIL.DAT are moved from the source sys$system to sys$specific. Later you will want to merge these with the ones in sys$common.

Set you default to sys$specific:[sysexe] and run authorize to modify the default DECNET account (and other DECNET accounts) to the common system disk accounts.

Copy any needed (at startup) directories and files from the source disk to the target disk.

Using the copy of your console media, edit the bootstrap command file to boot from the common system disk device.

Shutdown the node and try a conversational boot from the common system disk.

```
>>@CSDBOO.CMD
   .
   .
   .
SYSBOOT> SET /STARTUP SYS$SYSTEM:STARTUP.COM
SYSBOOT> SET STARTUPP1 "MIN"
SYSBOOT> CONTINUE
```

Login as system manager and invoke AUTOGEN

```
$ @SYS$UPDATE:AUTOGEN SAVPARAMS REBOOT
```

After the reboot, you should be running on the common system disk. If you have problems with startup files you may log in through one of the nodes running on the common system disk and modify the startup files for the new node.

Assuming that the boot, works start checking things. Logical names, queues and printer forms, etc. There will probably be a number of things that need fixing. The more attention given to making the

systems similar before attempting to add the new node the less work on these final fixes. After a period of stability (and backups) you may then want to start integrating the node specific stuff, such as SYSUAF.DAT into sys$common. A useful thing to do is to look for files duplicated in the separate node-specific roots.

2.5 SOME HELPFUL COMMAND FILES

You will need to move some selected files from the source node system disk to the common system disk. Here are some command files that have been created to aid in these steps. (modeled on some unknown author):

2.5.1 DIR_COMPARE.COM -

```
$! DIR_COMPARE.COM
$ set noon
$
$ write sys$output " "
$ write sys$output "Directory compare procedure."
$ Type sys$input

        (produces deleteduplicates.tmp)
The output file lists files that exist in only one of the
directories or exist in both, but are different internally
The output command file can delete the duplicates from
Directory  2

$ write sys$output " "
$
$ p2cmd = p2
$ p3cmd = p3
$
$ if p1 .eqs. "" then inquire p1 "Directory   1"
$ if p2 .eqs. "" then inquire p2 "Directory   2"
$
$ write sys$output " "
$
$ dir1 = P1
$ spec1 = dir1+"."
$ Next1 := TRUE
$
$ dir2 = P2
$ spec2 = dir2+"."
$ Next2 := TRUE
$
$ DeleteFiles := TRUE
$ DeleteFlag = ""
$
$ open/write cmdout deleteduplicates.tmp
$ comment = "$ ! "
$  write cmdout -
   comment," Directory Compare Procedure in VAX/VMS DCL ",-
   f$time()
$  write cmdout -
   comment," Produces DELETEDUPLICATES.TMP"
$  write cmdout -
```

```
   comment," to delete duplicates from directory  2"
$  write cmdout -
   comment," Comparing Directory  1 ",p1
$  write cmdout comment,"        to Directory  2 ",p2
$  write cmdout comment
$
$
$ F1:
$       if next1 .eqs. "FALSE"  then goto F2
$       file1 = f$search(spec1, 1)
$       nam = f$parse(file1,,,"NAME")+"          "
$       nam = f$extract(0,9,nam)
$       typ = f$parse(file1,,,"TYPE")+"    "
$       typ = f$extract(0,4,typ)
$       filename1 = nam+typ
$       if file1 .eqs. "" then filename1 :=
$       next1 := FALSE
$
$ F2:
$       if next2 .eqs. "FALSE"  then goto Check
$       file2 = f$search(spec2, 2)
$       nam = f$parse(file2,,,"NAME")+"          "
$       nam = f$extract(0,9,nam)
$       typ = f$parse(file2,,,"TYPE")+"    "
$       typ = f$extract(0,4,typ)
$       filename2 = nam+typ
$       if file2 .eqs. "" then filename2 :=
$       next2 := FALSE
$
$ Check:
$       if filename1 .nes. filename2  then goto CmpNam
$       if filename1 .eqs. ""  then goto done
$       Next1 := TRUE
$       Next2 := TRUE
$       dif/out=nl: 'file1' 'file2'
$       if $severity .ne. 1  then goto Dif
$
$       cmdlin := "$ DELETE/LOG " 'file2'
$       if DeleteFiles then write cmdout cmdlin
$       goto F1
$
$ Dif:
$       write cmdout comment,filename1," -- Different"
$       goto F1
$
$ CmpNam:
$       if filename1 .ges. filename2 then goto bigger
$       Next1 := TRUE
$       write cmdout -
        comment,file1," -- No file in directory   2"
$       goto F1
$
$ Bigger:
$       Next2 := TRUE
$       write cmdout -
```

```
        comment,file2," -- No file in directory   1"
$       goto F2
$
$ Done:
$       close cmdout
$       write sys$output "Done "
$       write sys$output " "
$       exit
```

==========

2.5.2 DUP_FILENAMES.COM -

```
$! DUP_FILENAMES.COM
$ set noon
$
$ write sys$output " "
$ write sys$output "Filename compare procedure."
$ Type sys$input

        (produces filenamesduplicates.tmp)
The output file lists files that exist in only one of
the directories.  The output command file can delete the
duplicates from Directory  2

$
$ p2cmd = p2
$ p3cmd = p3
$
$ if p1 .eqs. "" then inquire p1 "Directory   1"
$ if p2 .eqs. "" then inquire p2 "Directory   2"
$
$ write sys$output " "
$
$ dir1 = P1
$ spec1 = dir1+"."
$ Next1 := TRUE
$
$ dir2 = P2
$ spec2 = dir2+"."
$ Next2 := TRUE
$
$ DeleteFiles := TRUE
$ DeleteFlag = ""
$
$ open/write cmdout filenamesduplicates.tmp
$ comment = "$ ! "
$   write cmdout -
    comment," Directory Compare Procedure in VAX/VMS DCL "-
    ,f$time()
$   write cmdout -
    comment," Produces FILENAMESDUPLICATES.TMP"
$   write cmdout -
    comment," to delete duplicates from directory  2"
$   write cmdout -
    comment," Comparing Directory  1 ",p1
$   write cmdout comment,"        to Directory  2 ",p2
$   write cmdout comment
$
$
$ F1:
$       if next1 .eqs. "FALSE"  then goto F2
$       file1 = f$search(spec1, 1)
$       nam = f$parse(file1,,,"NAME")+"         "
$       nam = f$extract(0,9,nam)
$       typ = f$parse(file1,,,"TYPE")+"   "
$       typ = f$extract(0,4,typ)
```

```
$       filename1 = nam+typ
$       if file1 .eqs. "" then filename1 :=
$       next1 := FALSE
$
$ F2:
$       if next2 .eqs. "FALSE"  then goto Check
$       file2 = f$search(spec2, 2)
$       nam = f$parse(file2,,,"NAME")+"          "
$       nam = f$extract(0,9,nam)
$       typ = f$parse(file2,,,"TYPE")+"    "
$       typ = f$extract(0,4,typ)
$       filename2 = nam+typ
$       if file2 .eqs. "" then filename2 :=
$       next2 := FALSE
$
$ Check:
$       if filename1 .nes. filename2  then goto CmpNam
$       if filename1 .eqs. ""  then goto done
$       Next1 := TRUE
$       Next2 := TRUE
$
$       cmdlin := "$ DELETE/LOG " 'file2'
$       if DeleteFiles then write cmdout cmdlin
$       goto F1
$
$ CmpNam:
$       if filename1 .ges. filename2 then goto bigger
$       Next1 := TRUE
$       write cmdout -
        comment,file1," -- No file in directory   2"
$       goto F1
$
$ Bigger:
$       Next2 := TRUE
$       write cmdout -
        comment,file2," -- No file in directory   1"
$       goto F2
$
$ Done:
$       close cmdout
$       write sys$output "Done "
$       write sys$output " "
$       exit
```

==========

3 SERVED DISKS

In order for other cluster members to use these disks as though they were locally connected, you need to make the disks "served". However, there are some problems with served disks in a cluster environment. The rules are somewhat different for a Local Area VAX Cluster. See the next section for the LAVC differences.

Consider a CI based cluster with local (non-HSC) disks. In order to have served disks you will need to do the following steps:

1. Run CONFIGURE without doing an SYSGEN autoconfigure. However, the DEC standard STARTUP.COM precludes this. This can be done by running configure in SYCONFIG.COM.

2. Load the MSCP driver in your system startup file before mounting the served disks.

3. The auto-rebuild must be disabled, so that the rebuild may be postponed until after the disk is served.

As an example, first a SYCONFIG.COM template is shown.

```
$!      S Y C O N F I G . C O M
$!      ======================
$! Configure I/O data base for special, then standard devices
$! and their required drivers.
$! NOTE. This is a machine dependent procedure. It is tailored
$! to the specific hardware configuration, and is not
$! identical on all CPUs.
$!
$       SET NOON
!+
! Disable NETWORK messages as soon as possible to avoid a
! possible flood of such messages from another node while
! we are booting.
!-
$       DEFINE  SYS$COMMAND  OPA0:
$       REPLY/DISABLE=(NETWORK,SECURITY)
$       DEASSIGN  SYS$COMMAND
$!
$       RUN SYS$SYSTEM:SYSGEN
!!
!!  CAUTION:
!!      SYSGEN commands may be 'commented-out' with 1 (or 2)
!!      leading exclamation marks, but NOT with 3--with 3
!!      (or with 3n) the command is active.
!!
!
! AUTOCONFIGURE devices
!
AUTOCONFIGURE MB0
```

```
AUTOCONFIGURE CI0
   .
   .
   .
!
! Establish MSCP server for cluster-served disks
!
MSCP/HOSTS=3
!
EXIT
$!
!+
! Suppress the AUTOCONFIGURE ALL in STARTUP.COM, but
! execute the CONFIGURE process for cluster support.
!-
$       STARTUP$AUTOCONFIGUREALL== 0
$       @SYS$SYSTEM:STARTUP CONFIGURE
$!
$       EXIT    ! SYCONFIG.COM
```

When mounting the served disks the rebuild process must be postponed until after the disks are served. For instance:

```
$!
$!  Rebuild, then SERVE and MOUNT/CLUSTER system disk.
$!   NOTE: AUTO-REBUILD must have been suppressed by setting
$!         the sysgen parameter ACPREBLDSYSD to 0
$!         REBUILD must be postponed until after disk has
$!         been SERVED.
$!
$       SET DEVICE /SERVED  $1$DRA3:
$       MOUNT /SYSTEM /WRITE /NOASSIST /REBUILD /CLUSTER -
                                $1$DRA3:  disklabel logicalname
$!
```

4 LOCAL AREA VAX CLUSTER CONSIDERATIONS

There are some differences for local area VAX clusters.

4.1 CLUSTER SETUP OPERATIONS

The configuration of the boot node is like configuring a regular VAX cluster member. You may use the procedure in the first section for creating

the common system disk structures. After doing that install DECNET, if you do not already have it. Then install the Local Area VAXcluster Software key.
You will also need to execute the BOOTCONFIG.COM procedure on the boot node to define the cluster group number, password (use a long password -
as close to 31 characters as you can). You will also need to enter the SCSSYSTEMID and DECNET Node name for the boot node. A new cluster authorization file SYS$COMMON:[SYSEXE]CLUSTERAUTHORIZE.DAT will be added to your system.

The main difference is in adding satellite nodes. A satellite node may be
disk-less, or may have it's own local disk for paging and/or data storage.
A command procedure SATELLITECONFIG.COM is provided with the LAVC software
that can ADD, REMOVE, or MODIFY a satellite node. Note that to move a satellite's page and swap file it is recommended that you remove a satellite node and re-Add it.

If the satellite's local disk is to be served to other members of the cluster the procedure is different from that show in the section on served
disks. Instead you should:

1. Log onto the satellite

2. Add the line MSCPLOAD = 1 to sys$system:modparams.dat

3. Insure that the local disk(s) have a unique name through-out the
   whole cluster. Use $ SET VOLUME/LABEL=vol-label device:

4. Execute autogen with $ @SYS$UPDATE:AUTOGEN GETDATA REBOOT INITIAL

If you are adding a node that already has a local operating system you will
need to disable the automatic boot so that the boot actually takes place
from the host boot-node. If you also want to keep the ability to boot the
satellite separately as a stand-alone system again do the following:

```
$ RENAME SYS$SYSDEVICE:[000000]SYS0.DIR -
                SYS$SYSDEVICE:[000000]SYS1.DIR
$ SET FILE/REMOVE SYS$SYSDEVICE:[SYSEXE]SYSBOOT.EXE
```

Then to boot the satellite locally enter at the console prompt

```
>>> B/1000000
```

As one last little
reminder, often a cluster would like to have its own name.
Particularly if
the cluster is to be viewed as one DECNET address for mail, etc.
Choose
a network alias name (example clsnam) and create it using NCP.

```
$ RUN SYS$SYSTEM:NCP
$ DEFINE EXECUTOR ALIAS NODE clsnam
$ EXIT
```

Do the above on the boot node for a LAVC cluster. Do it on any of the members of a regular cluster, assuming that you have a common DECNET database. Not having a LAVC using a boot node that is also a regular cluster member I am not sure of what would happen. However, VMS 5.0 is supposed to support a mixed cluster, while VMS 4.x does not.

We have covered a number of topics. There are probably still
pitfalls that we have either not covered or discovered.
Good luck on your adventure.

LAWRENCE BERKELEY LABORATORY
TECHNICAL INFORMATION DEPARTMENT
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720