VMS Software

VSI OpenVMS

System Manager's Manual, Volume 2: Tuning, Monitoring, and Complex Systems

Operating System and Version: VSI OpenVMS x86-64 Version 9.2-1 or higher;
VSI OpenVMS IA-64 Version 8.4-1H1 or higher
VSI OpenVMS Alpha Version 8.4-2L1 or higher

VMS Software, Inc. (VSI)
Boston, Massachusetts, USA

System Manager’s Manual, Volume 2: Tuning, Monitoring, and Complex Systems

Copyright © 2025 VMS Software, Inc. (VSI), Boston, Massachusetts, USA

Legal Notice

Confidential computer software. Valid license from VSI required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

The information contained herein is subject to change without notice. The only warranties for VSI products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. VSI shall not be liable for technical or editorial errors or omissions contained herein.

HPE, HPE Integrity, HPE Alpha, and HPE Proliant are trademarks or registered trademarks of Hewlett Packard Enterprise.

Table of Contents

Preface

1. About VSI

VMS Software, Inc. (VSI) is an independent software company licensed by Hewlett Packard Enterprise to develop and support the OpenVMS operating system.

2. Intended Audience

This guide is intended for VSI OpenVMS system managers. VSI OpenVMS System Manager's Manual, Volume 2: Tuning, Monitoring, and Complex Systems assumes that readers are familiar with OpenVMS concepts and operation, and does not cover basic OpenVMS information.

3. Document Structure

This guide contains the following chapters and appendixes:

- *Chapter 1, "Managing System Parameters"* describes how to manage system parameters to fit your hardware configuration and your system's work load.
- *Chapter 2, "Managing Page, Swap, and Dump Files"* explains how to manage page, swap, and dump files.
- *Chapter 3, "Performance Considerations"* introduces basic concepts of performance management.
- *Chapter 4, "Managing File System Data Caches"* describes the caches that the Files-11 file system uses to cache data for ODS-2 and ODS-5 volumes.
- *Chapter 5, "Testing the System with UETP"* explains how to use user environment test package.
- *Chapter 6, "Getting Information About the System"* discusses setting up and maintaining system log files, maintaining error log files, and using system management utilities to monitor the system.
- *Chapter 7, "Tracking Resource Use"* describes how to find out how your system resources have been used.
- *Chapter 8, "OpenVMS Cluster Considerations"* describes concepts related to the OpenVMS Cluster environment.
- *Chapter 9, "Network Considerations"* discusses networking options for OpenVMS systems and their use.
- *Chapter 10, "Managing the Local Area Network (LAN) Software"* describes how to manage the LAN software on your system.
- *Chapter 11, "Managing InfoServer Systems"* describes InfoServer functions and InfoServer Client for OpenVMS software.
- *Chapter 12, "Managing the LAT Software"* describes how to manage the LAT software on your system.
- *Chapter 13, "Managing DECdtm Services"* explains how to use DECdtm services.

- *Chapter 14, "Managing Special Processing Environments"* describes how to set up and manage special processing environments.
- *Appendix A, "Files–11 Disk Structure"* explains disk terminology and disk concepts.
- *Appendix B, "Tables of Time Differential Factors"* shows the time differential factors (TDFs) of various locations in the world.
- *Appendix C, "VSI MIB Subagents Implemented on OpenVMS Alpha"* describes the VSI Server MIB and the VSI Cluster MIB.

4. Related Documents

For more information on the system management utilities, see the following documents:

- *VSI OpenVMS Guide to System Security*
- *VSI OpenVMS DCL Dictionary*
- *VSI OpenVMS System Manager's Manual*
- *VSI OpenVMS Programming Concepts Manual*
- *VSI OpenVMS Record Management Services Reference Manual*
- *VSI OpenVMS System Services Reference Manual*
- *VSI OpenVMS User's Manual*
- *OpenVMS VAX Device Support Manual* (archived)

5. VSI Encourages Your Comments

You may send comments or suggestions regarding this manual or any VSI document by sending electronic mail to the following Internet address: <docinfo@vmssoftware.com>. Users who have VSI OpenVMS support contracts through VSI can contact <support@vmssoftware.com> for help with this product.

6. OpenVMS Documentation

The full VSI OpenVMS documentation set can be found on the VMS Software Documentation webpage at https://docs.vmssoftware.com.

7. Typographical Conventions

The following conventions may be used in this manual:

Convention	Meaning
Ctrl/*x*	A sequence such as **Ctrl/***x* indicates that you must hold down the key labeled Ctrl while you press another key or a pointing device button.
PF1 *x*	A sequence such as PF1 *x* indicates that you must first press and release the key labeled PF1 and then press and release another key or a pointing device button.

Convention	Meaning
. . .	A horizontal ellipsis in examples indicates one of the following possibilities: ● Additional optional arguments in a statement have been omitted. ● The preceding item or items can be repeated one or more times. ● Additional parameters, values, or other information can be entered.
. . .	A vertical ellipsis indicates the omission of items from a code example or command format; the items are omitted because they are not important to the topic being discussed.
( )	In command format descriptions, parentheses indicate that you must enclose the options in parentheses if you choose more than one.
[ ]	In command format descriptions, brackets indicate optional choices. You can choose one or more items or no items. Do not type the brackets on the command line. However, you must include the brackets in the syntax for OpenVMS directory specifications and for a substring specification in an assignment statement.
[|]	In command format descriptions, vertical bars separate choices within brackets or braces. Within brackets, the choices are options; within braces, at least one choice is required. Do not type the vertical bars on the command line.
{ }	In command format descriptions, braces indicate required choices; you must choose at least one of the items listed. Do not type the braces on the command line.
bold text	This typeface represents the introduction of a new term. It also represents the name of an argument, an attribute, or a reason.
italic text	Italic text indicates important information, complete titles of manuals, or variables. Variables include information that varies in system output (Internal error *number*), in command lines (/PRODUCER= *name*), and in command parameters in text (where *dd* represents the predefined code for the device type).
UPPERCASE TEXT	Uppercase text indicates a command, the name of a routine, the name of a file, or the abbreviation for a system privilege.
`Monospace type`	Monospace type indicates code examples and interactive screen displays. In the C programming language, monospace type in text identifies the following elements: keywords, the names of independently compiled external functions and files, syntax summaries, and references to variables or identifiers introduced in an example.
-	A hyphen at the end of a command format description, command line, or code line indicates that the command or statement continues on the following line.
numbers	All numbers in text are assumed to be decimal unless otherwise noted. Nondecimal radixes—binary, octal, or hexadecimal—are explicitly indicated.

Chapter 1. Managing System Parameters

When your system is installed or upgraded, values of system parameters are automatically set by the command procedure SYS$UPDATE:AUTOGEN.COM (AUTOGEN), which is supplied by VSI. VSI recommends that you use AUTOGEN regularly to adjust the values for system parameters to fit your hardware configuration and your system's work load.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Converting your customized parameter settings for use with AUTOGEN	*Section 1.3, "Converting Your Customized Parameter Settings for Use with AUTOGEN"*
Modifying system parameter values with AUTOGEN (recommended method)	*Section 1.5, "Modifying System Parameters with AUTOGEN"*
Controlling AUTOGEN's parameter settings with MODPARAMS.DAT	*Section 1.5.1, "Controlling AUTOGEN's Parameter Settings with MODPARAMS.DAT"*
Automating AUTOGEN reports	*Section 1.6, "Automating AUTOGEN Reports"*
Managing system parameters with SYSMAN	*Section 1.7, "Managing System Parameters with SYSMAN"*
Managing system parameters with SYSGEN	*Section 1.8, "Managing System Parameters with SYSGEN"*
Managing system parameters with a conversational boot	*Section 1.9, "Modifying System Parameters with a Conversational Boot"*

This chapter explains the following concepts:

Concept	Section
System parameters	*Section 1.1, "Understanding System Parameters"*
Default, current, and active values of system parameters	*Section 1.1.1, "Default, CURRENT, and ACTIVE Values"*
Pages and pagelets	*Section 1.1.2, "Pages and Pagelets"*
The recommended method for changing system parameter values	*Section 1.2, "Recommended Method for Changing Parameter Values"*
The AUTOGEN command procedure	*Section 1.4, "Understanding the AUTOGEN Command Procedure"*
AUTOGEN feedback	*Section 1.4.1, "AUTOGEN Feedback"*
The AUTOGEN feedback report (AGEN$PARAMS.REPORT)	*Section 1.4.2, "Feedback Report (AGEN $PARAMS.REPORT)"*
AUTOGEN phases	*Section 1.4.3, "AUTOGEN Phases"*
The AUTOGEN parameter file (MODPARAMS.DAT)	*Section 1.4.4, "AUTOGEN Parameter File (MODPARAMS.DAT)"*

1.1. Understanding System Parameters

The system uses values for **system parameters** to control how the system functions. System parameters control a wide range of system functions, including but not limited to the following functions:

- Memory management
- Scheduling
- Security attributes
- System caches
- Windowing system choice

- Terminal configuration
- VAXcluster or OpenVMS Cluster system attributes

The *VSI OpenVMS System Management Utilities Reference Manual* lists and describes each system parameter.

Your distribution kit provides **default values** for system parameters to allow you to boot any supported configuration. When your system is installed or upgraded, the SYS$UPDATE:AUTOGEN.COM command procedure executes to evaluate your hardware configuration, estimate typical work loads, and adjust the values of system parameters as needed.

Each system parameter has associated minimum and maximum values that define the scope of allowable values.

Parameter Types

System parameters can be one or more of the following types:

Type	Description
Dynamic	The value of a dynamic system parameter can be modified while the system is active by changing the *active* value in memory. In contrast, if you change the value of a parameter that is not dynamic, you must change the *current* value stored in the parameter file, and you must reboot the system for the changed value to take effect. For information about active and current values, see *Section 1.1.1, "Default, CURRENT, and ACTIVE Values"*.
General	The value of a general parameter affects the creation and initialization of data structures at boot time.
Major	Major parameters are most likely to require modification.
Special	Special parameters are intended for use only by VSI. Change these parameters only if recommended by VSI personnel or in the installation guide or release notes of a VSI-supplied layered product.

Parameter Categories by Function

System parameters can be divided into the following categories, according to their function:

Category	Function
ACP	Parameters associated with file system caches and Files-11 XQP (extended QIO procedure) or ancillary control processes (ACPs).[1]
Cluster	Parameters that affect VAXcluster or OpenVMS Cluster system operation.
Job	Parameters that control jobs.
LGI	Parameters that affect login security.
Multiprocessing	Parameters associated with symmetric multiprocessing.
PQL	Parameters associated with process creation limits and quotas.
RMS	Parameters associated with OpenVMS Record Management Services (RMS).
SCS	Parameters that control system communication services (SCS) and port driver operation. The parameters that affect SCS operation have the prefix SCS.
SYS	Parameters that affect overall system operation.
TTY	Parameters associated with terminal behavior.

Category	Function
User-defined	The following parameters can be user-defined: USERD1 (dynamic) USERD2 (dynamic) USER3 USER4

[1]Many ACP parameters are applicable only when Files–11 On-Disk Structure Level 1 disks are mounted or when an ACP is specifically requested during a mount command. In versions of the operating system before VAX VMS Version 4.0, a separate process, the ancillary control process (ACP), performed file operations such as file opens, closes, and window turns. VAX VMS Version 4.0 introduced the XQP (extended QIO procedure), which allows every process on the system to perform these operations. For compatibility reasons, the names of the parameters have not changed.

1.1.1. Default, CURRENT, and ACTIVE Values

A system has several different sets of values for system parameters. The following table describes these values:

Value	Description
Default values	Values provided with the system to allow you to boot any supported configuration.
Current values	Values stored in the default parameter file on disk and used to boot the system. On VAX systems, the default parameter file is VAXVMSSYS.PAR. On Alpha systems, the default parameter file is ALPHAVMSSYS.PAR.
Active values	Values that are stored in memory and are used while the system is running. You can change the active value on a running system only for system parameters categorized as dynamic system parameters.
Values stored in other parameter files	For special purposes, you can create a parameter file other than the default parameter file that is used to store current values.

When the system boots, it reads the current values into memory, creating active values. An active value remains equal to the current value until you change either the active value or the current value.

When you execute the AUTOGEN command procedure through the SETPARAMS phase, it changes *current* values.

The System Management utility (SYSMAN) and the System Generation utility (SYSGEN) allow you to show and modify both *current* and *active* values. Use the USE and WRITE commands to specify which values you want to show or modify.

For more information about managing parameters with SYSMAN, see *Section 1.7, "Managing System Parameters with SYSMAN"*. For more information about managing parameters with SYSGEN, see *Section 1.8, "Managing System Parameters with SYSGEN"*.

1.1.2. Pages and Pagelets

On VAX systems, the operating system allocates and deallocates memory for processes in units called **pages**. A page on a VAX system is 512 bytes. Some system parameter values are allocated in units of pages.

On Alpha systems, some system parameter values are allocated in units of pages, while others are allocated in units of **pagelets**.

A page on an Alpha system can be 8 kilobytes (KB) (8192 bytes), 16 KB, 32 KB, or 64 KB. A page let is a 512-byte unit of memory. One Alpha pagelet is the same size as one VAX page. On an Alpha computer with a page size of 8 KB, 16 Alpha pagelets equal one Alpha page.

When reviewing parameter values, especially those parameters related to memory management, be sure to note the units required for each parameter. *Section 1.7.2, "Showing Parameter Values with SYSMAN"* and *Section 1.8.2, "Showing Parameter Values with SYSGEN"* explain how to show parameter values and their units of allocation.

1.2. Recommended Method for Changing Parameter Values

Many system parameters can affect other parameters and the performance of the system. For this reason, VSI recommends that you use the command procedure SYS$UPDATE:AUTOGEN.COM (AUTOGEN) to manage system parameters. For information about AUTOGEN, see *Section 1.4, "Understanding the AUTOGEN Command Procedure"*.

The System Management utility (SYSMAN) and the System Generation utility (SYSGEN) also allow you to manage system parameters. Although these utilities are not generally recommended for *changing* parameter values, you can use one of these utilities for the following reasons:

- To display system parameters and their values on a VAX or Alpha system
- To display system parameters and their values for systems in an OpenVMS Cluster environment
- To temporarily modify a single parameter that has little effect on other parameters

---

Caution

If you change a parameter value with SYSMAN or SYSGEN, the value you set will be overridden or reset to the default value when you run AUTOGEN. To retain the changes when you run AUTOGEN, you must add the parameter value to the AUTOGEN parameter file MODPARAMS.DAT. For more information, see *Section 1.5.1, "Controlling AUTOGEN's Parameter Settings with MODPARAMS.DAT"*.

If you currently use SYSMAN or SYSGEN to change parameters, and you have not added your customized parameter settings to MODPARAMS.DAT, follow the instructions in *Section 1.3, "Converting Your Customized Parameter Settings for Use with AUTOGEN"* before running AUTOGEN.

1.3. Converting Your Customized Parameter Settings for Use with AUTOGEN

VSI recommends that you use the AUTOGEN command procedure to tune your system. If you use the System Management utility (SYSMAN) or the System Generation utility (SYSGEN) to modify system parameter values, and you do not include these changes in the AUTOGEN parameter file MODPARAMS.DAT, these changes will be overridden the next time you run AUTOGEN.

If you used SYSMAN or SYSGEN to change parameter values in the past, use the following procedure to convert your parameter settings to work with AUTOGEN. This procedure explains how to add your customized parameter settings to MODPARAMS.DAT so they will be retained when you run AUTOGEN.

Before performing this task, you should understand AUTOGEN, feedback, and the AUTOGEN parameter file MODPARAMS.DAT, as explained in *Section 1.4, "Understanding the AUTOGEN Command Procedure"*.

1. Save the parameter values that the system is now using as follows:

   ```
   $ RUN SYS$SYSTEM:SYSMAN
   SYSMAN> PARAMETERS USE ACTIVE
   SYSMAN> PARAMETERS WRITE SYS$SYSTEM:nodename_PARAMS_CURRENT.PAR
   ```

2. Write a listing of the active parameter values to an ASCII file named *nodename*_PARAMS.OLD as follows:

   ```
   SYSMAN> PARAMETERS SHOW/ALL/OUTPUT=nodename_PARAMS.OLD
   SYSMAN> PARAMETERS SHOW/SPECIAL/OUTPUT=nodename_PARAMS_SPECIAL.OLD
   SYSMAN> EXIT
   $ APPEND nodename_PARAMS_SPECIAL.OLD nodename_PARAMS.OLD
   ```

   You will use this file in step 6.

3. Edit AUTOGEN's parameter file SYS$SYSTEM:MODPARAMS.DAT to define symbols to specify values for the following parameters:

   - Parameter values that are not calculated by AUTOGEN, such as SCSNODE and SCSSYSTEMID. Refer to the AUTOGEN description in the *VSI OpenVMS System Management Utilities Reference Manual* for a table of the parameters calculated by AUTOGEN.

   - Any parameter values that must be adjusted to suit your system work load, for example, GBLPAGES and GBLSECTIONS.

   To specify a value, define symbols using the format MIN_parameter, MAX_parameter, or ADD_parameter rather than specifying an explicit value. For example:

   ```
   $ EDIT SYS$SYSTEM:MODPARAMS.DAT

   SCSNODE = "MYNODE"  ! Not calculated by AUTOGEN
   SCSSYSTEMID = 10001  ! Not calculated by AUTOGEN
   MIN_GBLPAGES = 10000  ! Needed for MCS, BLISS32, and ADA
   MIN_GBLSECTIONS = 600  ! Needed for MCS, BLISS32, and ADA
   ```

   To help you track the changes you make in MODPARAMS.DAT, add comments to each line, preceded by an exclamation point (!). For information about defining symbols in MODPARAMS.DAT, see *Section 1.5.1, "Controlling AUTOGEN's Parameter Settings with MODPARAMS.DAT"*.

4. Run AUTOGEN, but do *not* reboot. Use one of the following commands, depending on your system:

   - If the system has run a typical work load for more than 24 hours since last booting:

     ```
     $ @SYS$UPDATE:AUTOGEN SAVPARAMS SETPARAMS FEEDBACK
     ```

     The SAVPARAMS phase collects feedback information about resource use on the running system; this information is used by AUTOGEN. This command creates a feedback report named SYS$SYSTEM:AGEN$PARAMS.REPORT, which tells you about peak resource use.

   - If you want to use a previously collected feedback file:

     ```
     $ @SYS$UPDATE:AUTOGEN GETDATA SETPARAMS FEEDBACK
     ```

If you start from the GETDATA phase, AUTOGEN does not collect current feedback.

- If this is a new system (that is, it has no feedback) or the system has had little activity since last boot (for example, over the weekend) so there is no valid feedback file:

  ```
  $ @SYS$UPDATE:AUTOGEN GETDATA SETPARAMS CHECK_FEEDBACK
  ```

  Use the CHECK_FEEDBACK parameter to let AUTOGEN determine whether the feedback is valid.

5. Write a listing of the new parameter values to an ASCII file as follows:

   ```
   $ RUN SYS$SYSTEM:SYSMAN
   SYSMAN> PARAMETERS USE CURRENT
   SYSMAN> PARAMETERS SHOW /ALL /OUTPUT=nodename_PARAMS.NEW
   SYSMAN> PARAMETERS SHOW /SPECIAL /OUTPUT=nodename_PARAMS_SPECIAL.NEW
   SYSMAN> EXIT
   $ APPEND nodename_PARAMS_SPECIAL.NEW nodename_PARAMS.NEW
   ```

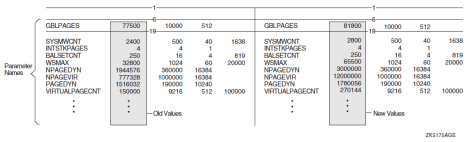
6. Compare the old and new parameter values as follows:

   ```
   $ DIFFERENCES/PARALLEL/OUTPUT=nodename_PARAMS.DIF/MATCH=5  -
   _$ nodename_PARAMS.OLD nodename_PARAMS.NEW
   ```

7. Print the differences file you created in step 6 (named in the format *nodename*_PARAMS.DIF). Print the file on a 132-column line printer to make the output easier to read.

8. Compare the numbers in the two columns following each parameter name column. The left column shows the old value; the right column shows the new value. *Figure 1.1, "Old and New Parameter Values"* illustrates sample output.

   Figure 1.1. Old and New Parameter Values

9. Make any adjustments in MODPARAMS.DAT using symbols prefixed by MIN_, MAX_, or ADD_. For example, if AUTOGEN calculated a smaller value for GBLPAGES, you might specify a minimum value for this parameter as follows:

   ```
   MIN_GBLPAGES = 10000
   ```

   If you originally specified a parameter value in MODPARAMS.DAT (in step 3) but the parameter has not been changed, verify the following data:

   - The parameter name is spelled correctly and completely (not abbreviated). In MODPARAMS.DAT, AUTOGEN sees parameter names as symbol assignments. AUTOGEN cannot equate a symbol to the corresponding system parameter unless it is spelled correctly. Look in AGEN$FEEDBACK.REPORT for any error messages AUTOGEN might have written.

   - The value is correct: count the digits and make sure no commas are present.

   - The parameter occurs only once in MODPARAMS.DAT.

- The parameter is not commented out.

  For most parameters, if the new value is greater than the old value, you can accept AUTOGEN's setting. If the new value is less than the old value, VSI recommends that you retain the old value because the system may not have been using that resource when running AUTOGEN. For example, you might have used SYSMAN to increase GBLPAGES to 10, 000 to accommodate layered products, but have not specified that change in MODPARAMS.DAT. AUTOGEN might calculate that the system needs only 5000 global pages. When you reboot after running AUTOGEN, not all of your layered products may be installed, and you might receive the system message GPTFULL, "global page table full, " indicating that the system needs more GBLPAGES.

10. Repeat from step 3 until you are satisfied with the new parameter values.

    If necessary, make further changes in MODPARAMS.DAT, run AUTOGEN again, and verify the changes as before. Usually after this second pass of AUTOGEN, the parameter values will be stable and you can then reboot.

11. Reboot. When you reboot, the system will use the new parameter values. Using AUTOGEN to reboot or rebooting right away is not necessary. However you must reboot before the system uses the new parameter values. If the system does not boot, perform a conversational boot and use the backup parameter file you created in step 1:

    ```
    SYSBOOT> USE SYS$SYSTEM:nodename_PARAMS_CURRENT.PAR
    SYSBOOT> CONTINUE
    ```

    When you enter the CONTINUE command, the system boots with the parameter values you saved before running AUTOGEN.

    After the system has booted, if you want to use the old parameter values you can enter the following commands:

    ```
    $ RUN SYS$SYSTEM:SYSMAN
    SYSMAN> PARAMETERS USE SYS$SYSTEM:nodename_PARAMS_CURRENT.PAR
    SYSMAN> PARAMETERS WRITE CURRENT
    SYSMAN> EXIT
    ```

12. Run AUTOGEN using feedback regularly to ensure that the resources of your system match your system work load. For information about running AUTOGEN using feedback, see *Section 1.5, "Modifying System Parameters with AUTOGEN"*

1.4. Understanding the AUTOGEN Command Procedure

The AUTOGEN command procedure, SYS$UPDATE:AUTOGEN.COM, is provided on your distribution kit, and runs automatically when your system is installed or upgraded to set appropriate values for system parameters. In addition, VSI recommends that you run AUTOGEN when you want to reset values for system parameters or to resize page, swap, and dump files. The new values and file sizes take effect the next time the system boots.

AUTOGEN only calculates certain significant system parameters. Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for more information.

When to Run AUTOGEN

VSI recommends running AUTOGEN in the following circumstances:

- During a new installation or upgrade. (This happens automatically as part of the installation or upgrade procedure.)
- Whenever your work load changes significantly.
- When you add an optional (layered) software product. See the specific product documentation for installation requirements. Certain layered products might require you to execute AUTOGEN to adjust parameter values and page and swap file sizes. (For information about using AUTOGEN to modify page and swap files, see *Section 2.15.1, "Using AUTOGEN (Recommended Method)"*.)
- When you install images with the /SHARED attribute; the GBLSECTIONS and GBLPAGES parameters might need to be increased to accommodate the additional global pages and global sections consumed.
- On a regular basis to monitor changes in your system's work load. You can automate AUTOGEN to regularly check feedback and recommend system parameter changes. *Section 1.6, "Automating AUTOGEN Reports"* describes a batch-oriented command procedure that runs AUTOGEN in feedback mode on a regular basis and automatically sends the feedback report to an appropriate Mail account.
- Periodically to provide adequate swapping file space. Use the FEEDBACK option and make sure the system has been up long enough (at least 24 hours) and that the load is typical. Also, make sure the SYS$SYSTEM:MODPARAMS.DAT file does not contain a hardcoded SWAPFILE value, which prevents AUTOGEN from correctly sizing the swapping files.

AUTOGEN Operations

AUTOGEN executes in phases. Depending on which phases you direct it to execute, AUTOGEN performs some or all of the following operations:

- Collects the following types of data:
  - Feedback (from the running system)
  - The hardware configuration (from the system)
  - Parameter requirements supplied by you (from MODPARAMS.DAT)
  - Parameter requirements supplied by VSI
- Calculates appropriate new values for significant system parameters (listed in the *VSI OpenVMS System Management Utilities Reference Manual*)
- Creates a new installed image list
- Calculates the sizes of system page, swap, and dump files
- Adjusts the sizes of system page, swap, and dump files values of system parameter values, if necessary

- Optionally shuts down and reboots the system

Invoking AUTOGEN

To invoke AUTOGEN, enter a command in the following format at the DCL prompt:

```
@SYS$UPDATE:AUTOGEN [start-phase] [end-phase] [execution-mode]
```

where:

start-phase	Is the phase where AUTOGEN is to begin executing. *Section 1.4.3, "AUTOGEN Phases"* lists the AUTOGEN phases.
end-phase	Is the phase where AUTOGEN is to complete executing. *Section 1.4.3, "AUTOGEN Phases"* lists the AUTOGEN phases.
execution-mode	Is one of the following modes: • FEEDBACK—Use feedback. • NOFEEDBACK—Do not use feedback. • CHECK_FEEDBACK—Use feedback if it is valid. If feedback is invalid, ignore it, but continue executing through the end phase. • Blank (no execution mode specified)---Use feedback if it is valid. If it is not valid, quit before making any modifications.

For detailed information about invoking AUTOGEN, and the command line parameters you can specify, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

Controlling AUTOGEN Operations

Table 1.1, " Controlling AUTOGEN" summarizes the methods for controlling AUTOGEN behavior.

Table 1.1. Controlling AUTOGEN

To Control...	Use This Method...
Which operations AUTOGEN is to perform	Specify a start phase and an end phase when you invoke AUTOGEN.
Parameter values set by AUTOGEN	Specify values in the AUTOGEN parameter file MODPARAMS.DAT. Periodically examine the results of calculations that AUTOGEN makes to determine whether AUTOGEN has drawn the correct conclusions about your hardware configuration and to be sure the system parameter values are appropriate for your workload requirements. If the values are not appropriate, adjust them by specifying desired values in MODPARAMS.DAT. For more information on MODPARAMS.DAT, see *Section 1.4.4, "AUTOGEN Parameter File (MODPARAMS.DAT)"*.
AUTOGEN's use of feedback information	Specify an execution mode when you invoke AUTOGEN.AUTOGEN can often improve system performance by using dynamic feedback gathered from the running system. However, feedback information is not always valid or appropriate. For more information, see *Section 1.4.1, "AUTOGEN Feedback"*.

1.4.1. AUTOGEN Feedback

AUTOGEN feedback minimizes the need for you to modify parameter values or system file sizes. Instead, feedback allows AUTOGEN to automatically size the operating system based on your actual work load. **Sizing** is the process of matching the allocation of system resources (memory and disk space) with the workload requirements of your site.

Feedback is information, continuously collected by the operating system executive, about the amount of various resources the system uses to process its work load. The information is collected when exception events occur, so the collection does not affect system performance. When run in *feedback mode*, AUTOGEN analyzes this information and adjusts any related parameter values.

Note

When running AUTOGEN after making a major configuration change, specify **nofeedback** to assure the use of initial AUTOGEN settings. See *Section 1.4, "Understanding the AUTOGEN Command Procedure"*.

AUTOGEN feedback affects the following resources (for a complete list of the affected system parameters, refer to the *VSI OpenVMS System Management Utilities Reference Manual*):

- Nonpaged pool
- Paged pool
- Lock resources
- Number of processes
- Global pages
- Global sections
- File system caches
- System logical name table sizes
- Page files
- Swap files

Feedback is gathered during AUTOGEN's SAVPARAMS phase and is written to the file SYS $SYSTEM:AGEN$FEEDBACK.DAT. This file is then read during the GETDATA phase. (See *Section 1.4.3, "AUTOGEN Phases"* for more information about AUTOGEN phases.)

Feedback is useful only if it accurately reflects the system's normal work load. For this reason, AUTOGEN performs some basic checks on the feedback and issues a warning message for either of the following conditions:

- The system has been up for less than 24 hours.
- The feedback is over 30 days old.

Whenever you modify the system (for example, a hardware upgrade, a change in the number of users, an optional product installation), you should operate in the new system environment for a period of time, and then execute AUTOGEN again starting from the SAVPARAMS phase.

On VAX systems, you can define the logical name AGEN$FEEDBACK_REQ_TIME to specify, in hours, a minimum age required for feedback. For more information, see *Section 1.5.2, "Specifying a Minimum Required Age for Feedback(VAX Only)"*.

When AUTOGEN runs, it displays whether feedback is used, as follows:

```
Feedback information was collected on 21-JAN-2016 14:00:08.53
Old values below are the parameter values at the time of collection.
The feedback data is based on 21 hours of up time.
Feedback information will be used in the subsequent calculations
```

1.4.2. Feedback Report (AGEN$PARAMS.REPORT)

Decides whether to use the system parameter values and system file sizes calculated by AUTOGEN. To help in your decision making, AUTOGEN generates a report file (SYS$SYSTEM:AGEN $PARAMS.REPORT) that includes the following information:

- All parameters and system files directly affected by the feedback
- Current values
- New values
- The feedback used in each parameter calculation
- Any user- or VSI-supplied modifications found in MODPARAMS.DAT
- Any advisory or warning messages displayed during AUTOGEN's operations
- On VAX systems, any user- or VSI-supplied modifications found in VMSPARAMS.DAT
- On Alpha systems, the parameters found during the GENPARAMS phase

Example 1.1, "Sample AUTOGEN Feedback Report" shows the contents of a sample AUTOGEN feedback report for a VAX system. On Alpha systems, the feedback report is similar but not identical to this example.

Suppressing Informational Messages

To suppress the display of informational messages, define the AGEN $REPORT_NO_INFORMATIONALS logical to TRUE. Messages are entered in SYS$SYSTEM:AGEN $PARAMS.REPORT regardless of the value of AGEN$REPORT_NO_INFORMATIONALS.

Filtering DCL Statements from Your Report

The feedback report will contain DCL statements in MODPARAMS.DAT that are not simple assignments to system parameters or the ADD_, MAX_, or MIN_ extensions. To filter these statements out of the report, begin each statement in MODPARAMS.DAT with a dollar sign ($).

Example 1.1. Sample AUTOGEN Feedback Report

```
AUTOGEN Parameter Calculation Report on node: NODE22
  This information was generated at 23-APR-2016 01:45:47.87
  AUTOGEN was run from GETDATA to TESTFILES using FEEDBACK

** No changes will be done by AUTOGEN **
   The values given in this report are what AUTOGEN would
    have set the parameters to.
```

```
Processing Parameter Data files
-------------------------------

** WARNING ** - The system was up for less than 24 hours when the feedback
information was recorded. This could result in feedback information
that does not accurately reflect your typical work load.

Including parameters from: SYS$SYSTEM:MODPARAMS.DAT
The following was detected within MODPARAMS.DAT
   Please review immediately.

** INFORMATIONAL ** - Multiple MIN values found for MIN_CHANNELCNT.
 Using MODPARAMS value (550) which is superseding OpenVMS value (255)

** INFORMATIONAL ** - Multiple MIN values found for MIN_SWPOUTPGCNT.
 Using MODPARAMS value (1000) which is superseding OpenVMS value (500)

** INFORMATIONAL ** - Multiple MIN values found for MIN_PQL_DWSEXTENT.
 Using MODPARAMS value (11000) which is superseding OpenVMS value (1024)

** INFORMATIONAL ** - Multiple MIN values found for MIN_PQL_MWSEXTENT.
 Using MODPARAMS value (11000) which is superseding OpenVMS value (1024)
Feedback information was collected on 22-APR-2016 14:00:07.70
  Old values below are the parameter values at the time of collection.
  The feedback data is based on 13 hours of up time.
  Feedback information will be used in the subsequent calculations


Parameter information follows:
------------------------------

MAXPROCESSCNT parameter information:
 Feedback information.
    Old value was 100, New value is 80
    Maximum Observed Processes: 52

Information about VMS executable image Processing:

 Processing SYS$MANAGER:VMS$IMAGES_MASTER.DAT


GBLPAGFIL parameter information:
 Override Information - parameter calculation has been overridden.
    The calculated value was 1024.  The new value is 6024.
    GBLPAGFIL has been increased by 5000.
    GBLPAGFIL is not allowed to be less than 6024.


GBLPAGES parameter information:
 Feedback information.
    Old value was 43300, New value is 50000
    Peak used GBLPAGES: 36622
    Global buffer requirements: 6024


GBLSECTIONS parameter information:
 Feedback information.
    Old value was 400, New value is 400
```

```
    Peak used GBLSECTIONS: 294
 Override Information - parameter calculation has been overridden.
    The calculated value was 350.  The new value is 400.
    GBLSECTIONS is not allowed to be less than 400.

LOCKIDTBL parameter information:
 Feedback information.
    Old value was 2943, New value is 3071
    Current number of locks: 1853
    Peak number of locks: 3200

LOCKIDTBL_MAX parameter information:
 Feedback information.
    Old value was 65535, New value is 65535

RESHASHTBL parameter information:
 Feedback information.
    Old value was 1024, New value is 1024
    Current number of resources: 957

MSCP_LOAD parameter information:
 Override Information - parameter calculation has been overridden.
    The calculated value was 1.  The new value is 0.
    MSCP_LOAD has been disabled by a hard-coded value of 0.

MSCP_BUFFER parameter information:
 Feedback information.
    Old value was 128, New value is 128
    MSCP server I/O rate: 0 I/Os per 10 sec.
    I/Os that waited for buffer space: 0
    I/Os that fragmented into multiple transfers: 0

SCSCONNCNT parameter information:
 Feedback information.
    Old value was 5, New value is 5
    Peak number of nodes: 1
    Number of CDT allocation failures: 0

SCSRESPCNT parameter information:
 Feedback information.
    Old value was 300, New value is 300
    RDT stall count: 0

SCSBUFFCNT parameter information:
 Feedback information.
    Old value was 512, New value is 512
    CIBDT stall count: 0


NPAGEDYN parameter information:
 Feedback information.
    Old value was 686592, New value is 783360
    Maximum observed non-paged pool size: 815616 bytes.
    Non-paged pool request rate: 47 requests per 10 sec.


LNMSHASHTBL parameter information:
 Feedback information.
```

```
    Old value was 1024, New value is 1024
    Current number of shareable logical names: 1194


ACP_DIRCACHE parameter information:
 Feedback information.
    Old value was 88, New value is 88
    Hit percentage: 99%
    Attempt rate: 0 attempts per 10 sec.

ACP_DINDXCACHE parameter information:
 Feedback information.
    Old value was 25, New value is 25
    Hit percentage: 97%
    Attempt rate: 1 attempts per 10 sec.

ACP_HDRCACHE parameter information:
 Feedback information.
    Old value was 88, New value is 106
    Hit percentage: 98%
    Attempt rate: 17 attempts per 10 sec.

ACP_MAPCACHE parameter information:
 Feedback information.
    Old value was 8, New value is 8
    Hit percentage: 2%
    Attempt rate: 4 attempts per 10 sec.

PAGEDYN parameter information:
 Feedback information.
    Old value was 521728, New value is 542208
    Current paged pool usage: 304160 bytes.
    Paged pool request rate: 1 requests per 10 sec.

PFRATL parameter information:
 Override Information - parameter calculation has been overridden.
    The calculated value was 0.  The new value is 1.
    PFRATL has been disabled by a hard-coded value of 1.

WSDEC parameter information:
 Override Information - parameter calculation has been overridden.
    The calculated value was 35.  The new value is 19.
    WSDEC has been disabled by a hard-coded value of 19.

MPW_LOLIMIT parameter information:
 Override Information - parameter calculation has been overridden.
    The calculated value was 120.  The new value is 2100.
    MPW_LOLIMIT is not allowed to be less than 2100.

MPW_HILIMIT parameter information:
 Override Information - parameter calculation has been overridden.
    The calculated value was 1310.  The new value is 4500.
    MPW_HILIMIT is not allowed to be less than 4500.

LONGWAIT parameter information:
 Override Information - parameter calculation has been overridden.
    The calculated value was 30.  The new value is 10.
    LONGWAIT has been disabled by a hard-coded value of 10.
```

```
WSMAX parameter information:
 Override Information - parameter calculation has been overridden.
    The calculated value was 8200.  The new value is 12000.
    WSMAX is not allowed to be less than 12000.


PROCSECTCNT parameter information:
 Override Information - parameter calculation has been overridden.
    The calculated value was 32.  The new value is 40.
    PROCSECTCNT is not allowed to be less than 40.


PQL_DWSEXTENT parameter information:
 Override Information - parameter calculation has been overridden.
    The calculated value was 400.  The new value is 11000.
    PQL_DWSEXTENT is not allowed to be less than 11000.


PQL_MWSEXTENT parameter information:
 Override Information - parameter calculation has been overridden.
    The calculated value was 2048.  The new value is 11000.
    PQL_MWSEXTENT is not allowed to be less than 11000.

VAXCLUSTER parameter information:
 Override Information - parameter calculation has been overridden.
    The calculated value was 1.  The new value is 0.
    VAXCLUSTER has been disabled by a hard-coded value of 0.

Page, Swap, and Dump file calculations

  Page and Swap file calculations.

PAGEFILE1_SIZE parameter information:
 Feedback information.
    Old value was 45200, New value is 50500
    Maximum observed usage: 25265
 PAGEFILE1_SIZE will be modified to hold 50500 blocks

PAGEFILE2_SIZE parameter information:
 Feedback information.
    Old value was 154000, New value is 194400
    Maximum observed usage: 97175
 PAGEFILE2_SIZE will be modified to hold 194400 blocks

 ** WARNING ** - The disk on which PAGEFILE2 resides would be
    over 95% full if it were modified to hold 194400 blocks.
    NODE22$DKA300:[SYSTEM_FILES]PAGEFILE.SYS will not be modified.
    NODE22$DKA300:[SYSTEM_FILES]PAGEFILE.SYS will remain at 154002 blocks.

SWAPFILE1_SIZE parameter information:
 Feedback information.
    Old value was 15000, New value is 15000
    Maximum observed usage: 14280
 Override Information - parameter calculation has been overridden.
    The calculated value was 21400.  The new value is 15000.
    SWAPFILE1_SIZE is not allowed to exceed 15000.
```

```
 SWAPFILE1 will not be modified.

SWAPFILE2_SIZE parameter information:
 Feedback information.
    Old value was 50000, New value is 26300
    Maximum observed usage: 1680
 SWAPFILE2_SIZE will be modified to hold 26300 blocks

 ** WARNING ** - The disk on which SWAPFILE2 resides would be
    over 95% full if it were modified to hold 26300 blocks.
    NODE22$DKA300:[SYSTEM_FILES]SWAPFILE.SYS will not be modified.
    NODE22$DKA300:[SYSTEM_FILES]SWAPFILE.SYS will remain at 50001 blocks.


  Dumpfile calculations:

 No dump file modifications would have been made.
 Dumpfile will remain at 34116 blocks.
```

1.4.3. AUTOGEN Phases

When you invoke AUTOGEN, you specify a start phase and an end phase for AUTOGEN to execute. AUTOGEN executes all phases from the start phase to the end phase. Depending on the start phase and end phase you specify, AUTOGEN can execute any of the following phases, in the order shown in *Table 1.2, "AUTOGEN Phases"*.

Table 1.2. AUTOGEN Phases

Phase	Description
SAVPARAMS	Saves dynamic feedback from the running system.
GETDATA	Collects all data to be used in AUTOGEN calculations.
GENPARAMS	Generates new system parameters; creates the installed image list.
TESTFILES	Displays the system page, swap, and dump file sizes calculated by AUTOGEN (cannot be used as a start phase).
GENFILES	Generates new system page, swap, and dump files if appropriate (cannot be used as a start phase).
SETPARAMS	Runs SYSMAN to set the new system parameters in the default parameter file, saves the original parameters, and generates a new parameter file, AUTOGEN.PAR. On VAX systems, the default parameter file is VAXVMSSYS.PAR. The original parameters are saved in the file VAXVMSSYS.OLD. On Alpha systems, the default parameter file is ALPHAVMSSYS.PAR. The original parameters are saved in the file ALPHAVMSSYS.OLD.
SHUTDOWN	Prepares the system to await a manual reboot.
REBOOT	Automatically shuts down and reboots the system.
HELP	Displays help information to the screen.

For detailed information about each AUTOGEN phase and the files affected by each phase, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

1.4.4. AUTOGEN Parameter File (MODPARAMS.DAT)

AUTOGEN reads a parameter file named MODPARAMS.DAT during the GETDATA phase. You can add commands to this file to control the system parameter values and file sizes that AUTOGEN sets. You can use MODPARAMS.DAT to perform the following actions:

Operation	For More Information
Increase the value of any numeric system parameter	*Section 1.5.1.1, "Increasing a Value with the ADD_ Prefix"*
Set a minimum value for a numeric system parameter	*Section 1.5.1.2, "Specifying a Minimum Value with the MIN_ Prefix"*
Set a maximum value for a numeric system parameter	*Section 1.5.1.3, "Specifying a Maximum Value with the MAX_ Prefix"*
Specify an absolute value for a system parameter	*Section 1.5.1.4, "Specifying an Absolute Value"*
Include an external parameter file	*Section 1.4.4, "AUTOGEN Parameter File (MODPARAMS.DAT)"*
Specify sizes of page, swap, and dump files	*Section 2.15.1.2, "Controlling the Size of System Page, Swap, and Dump Files in MODPARAMS.DAT"*
Define the number of VAXcluster nodes (VAX specific)	*Section 1.5.1.5, " Defining the Number of VAXcluster Nodes(VAX Only)"*
Define the number of Ethernet adapters (VAX specific)	*Section 1.5.1.6, "Defining the Number of Ethernet Adapters(VAX Only)"*
Preset parameter values before adding memory (VAX specific)	*Section 1.5.1.7, "Presetting Parameter Values Before Adding Memory(VAX Only)"*
Specify an alternate default startup command procedure	*VSI OpenVMS System Manager's Manual, Volume 1: Essentials*

To help track changes you make to MODPARAMS.DAT, make sure you add comments, preceded by the exclamation point (!), each time you change the file.

Caution

The recommended method of changing system parameters and system file sizes is to edit MODPARAMS.DAT to include parameter settings. If you change a system parameter value or file size using SYSMAN, SYSGEN, or a conversational boot, and you do not specify the value in MODPARAMS.DAT, AUTOGEN will recalculate the value or file size the next time it runs. For more information, see *Section 1.5.1, "Controlling AUTOGEN's Parameter Settings with MODPARAMS.DAT"*.

Example

The following example shows the contents of a sample MODPARAMS.DAT file:

```
!
! ***************** A Sample MODPARAMS.DAT for Node NODE22 ***************
!
! MODPARAMS.DAT for "NODE22"
! REVISED: 04/29/00 -CHG- Upped GBLPAGES to account for ADA.
```

```
!
SCSNODE         = "NODE22"      ! This is not calculated by AUTOGEN.
SCSSYSTEMID     = 19577         ! This is not calculated by AUTOGEN.
TTY_DEFCHAR2    = %X0D34        ! This is not calculated by AUTOGEN.
ADD_ACP_DIRCACHE= 150           ! Hit rate was only 65% on directory cache.
MIN_PAGEDYN     = 500000        ! PAGEDYN must be at least 1/2 Mbyte to
                                ! account for a large number of logical
 names.
!

MAX_PAGEFILE1_SIZE  = 15000     ! Maximum size for primary page.
MAX_SWAPFILE        =  5000     ! Maximum size for swap file space.
MAX_DUMPFILE        = 32768     ! Maximum size for dump file space.

ADD_GBLPAGES    = 425+507+157   ! Account for MCS, BLISS32 and ADA.
ADD_GBLSECTIONS = 4 + 5 + 2     ! Account for MCS, BLISS32 and ADA.
VIRTUALPAGECNT  = 144264        ! So that we can read MONSTR's 68Mb dumps.
!
! end of MODPARAMS.DAT for NODE22
```

1.5. Modifying System Parameters with AUTOGEN

The recommended method of modifying system parameters is to execute AUTOGEN in two passes, as follows:

1. First pass – Execute AUTOGEN using the following command:

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

   This command instructs AUTOGEN to perform the following actions:

   - Save the current feedback
   - Gather all the information required for the calculations
   - Calculate the system parameter values
   - Generate the feedback report
   - Write the information to SETPARAMS.DAT
   - Compute the sizes of page, swap, and dump files; log the sizes in the report file AGEN $PARAMS.REPORT.

   Review the input to the calculations (PARAMS.DAT), the output from the calculations (SETPARAMS.DAT), and the report generated (AGEN$PARAMS.REPORT). If you are not satisfied with the parameter settings, modify the parameter values by editing MODPARAMS.DAT as explained in *Section 1.5.1, "Controlling AUTOGEN's Parameter Settings with MODPARAMS.DAT"*. If you are not satisfied with the file sizes, modify the sizes as explained in *Section 2.15, "Creating and Modifying Page, Swap, and Dump Files"*. Then reexecute AUTOGEN from the GETDATA phase.

   When you are satisfied with the contents of SETPARAMS.DAT, go to step 2.

2. Second pass – Execute AUTOGEN a second time using the following command:

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

This AUTOGEN command runs SYSMAN to update the new system parameter values and installs them on the system when it is rebooted.

1.5.1. Controlling AUTOGEN's Parameter Settings with MODPARAMS.DAT

If, after examining the AGEN$PARAMS.REPORT or SETPARAMS.DAT file, you decide to correct hardware configuration data or modify system parameter values chosen by AUTOGEN, edit the MODPARAMS.DAT file as described in this section to manually specify parameter values.

Caution

Always edit MODPARAMS.DAT to specify values for parameters. Do not edit PARAMS.DAT; modifying the contents of this file might prevent AUTOGEN from operating correctly.

For information about editing MODPARAMS.DAT to control sizes of page, swap, and dump files, see *Section 2.15.1.2, "Controlling the Size of System Page, Swap, and Dump Files in MODPARAMS.DAT"*.

You can define symbols in MODPARAMS.DAT using the following formats to control parameter values:

Control Method	**Symbol Format**	**For More Information**
Increase a value by a specified amount	ADD_*	*Section 1.5.1.1, "Increasing a Value with the ADD_ Prefix"*
Specify a minimum value	MIN_*	*Section 1.5.1.2, "Specifying a Minimum Value with the MIN_ Prefix"*
Specify a maximum value	MAX_*	*Section 1.5.1.3, "Specifying a Maximum Value with the MAX_ Prefix"*
Specify an absolute value	Parameter name	*Section 1.5.1.4, "Specifying an Absolute Value"*

When defining symbols in MODPARAMS.DAT, make sure of the following data:

- The value is correct and valid for the parameter. Count the digits. Do not use commas.
- The symbol occurs only once in MODPARAMS.DAT.
- The symbol value is not commented out.
- The symbol name is spelled correctly and completely (not abbreviated).

Caution

When AUTOGEN reads MODPARAMS.DAT or any other parameter file, it checks to determine if the symbol names specified in the file are valid. If they are not, AUTOGEN writes a warning message to AGEN$PARAMS.REPORT. However, AUTOGEN checks only the symbol name; it does not check the validity of the value specified for the symbol.

If a value is invalid, the line is *not* ignored. AUTOGEN attempts to use the specified value.

A symbol is not checked if it is specified in a line that contains a DCL expression other than the equal sign (=). For example, AUTOGEN does not check the validity of a symbol name specified in a line with the DCL IF statement. Instead, AUTOGEN writes a warning message to AGEN$PARAMS.REPORT.

To help track changes you make to MODPARAMS.DAT, make sure you add comments preceded by an exclamation point (!) each time you change the file.

1.5.1.1. Increasing a Value with the ADD_ Prefix

Use the ADD_ prefix to increase the value of any numeric parameter. The new values are updated in subsequent AUTOGEN calculations during the GENPARAMS phase. The following example demonstrates the use of the ADD_ prefix:

```
ADD_GBLPAGES=500
ADD_NPAGEDYN=10000
```

An ADD_ parameter record for a parameter that AUTOGEN calculates will add the value to AUTOGEN's calculations. An ADD_ parameter record for a parameter that AUTOGEN does not calculate will add the value to the parameter's default (not current) value. (Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for a table of parameters affected by AUTOGEN.)

Note

The ADD_ value is added to the calculated value once, and does not accumulate with successive runs for feedback calculations.

Typically, you would not use the ADD_ prefix for modifying parameters that are calculated by the feedback mechanism, because the feedback results should accurately reflect your work load. However, if you do use the ADD_ prefix with feedback, AUTOGEN adds a value only once if AUTOGEN is run to the SETPARAMS phase or beyond. To maintain a minimum level above AUTOGEN's calculation, use the MIN_ prefix.

1.5.1.2. Specifying a Minimum Value with the MIN_ Prefix

Use the MIN_ prefix if you do not want AUTOGEN to set a parameter below a specified value. MIN_ refers to the minimum value to which a parameter can be set by AUTOGEN. The following example sets the minimum value to 400, 000:

```
MIN_PAGEDYN = 400000
```

1.5.1.3. Specifying a Maximum Value with the MAX_ Prefix

Use the MAX_ prefix if you do not want AUTOGEN to set a parameter above a specified value. MAX_ refers to the maximum value to which a parameter can be set by AUTOGEN. The following example sets the maximum value to 400, 000:

```
MAX_PAGEDYN = 400000
```

1.5.1.4. Specifying an Absolute Value

Use this method to specify a value for a parameter that AUTOGEN does not calculate. (Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for a table of the system parameters modified in AUTOGEN calculations.)

Note

VSI strongly recommends that you use this method only for parameters that describe the system environment (for example, SCSNODE and SCSSYSTEMID). For the parameters that AUTOGEN calculates, specifying a value with this method disables AUTOGEN's calculations. Instead of specifying an absolute value, use one of the following methods:

- Specify a minimum value with the MIN_ prefix
- Specify a maximum value with the MAX_ prefix
- Increase the value with the ADD_ prefix

To specify an absolute parameter value, add an assignment statement in the following format to MODPARAMS.DAT:

```
parameter = parameter-value ! comment
```

For example, the following command assigns the node name BIGVAX to the SCSNODE parameter:

```
SCSNODE = "BIGVAX"   ! the node name
```

1.5.1.5. Defining the Number of VAXcluster Nodes (VAX Only)

In a VAXcluster environment, use the NUM_NODES symbol to prevent temporary changes in VAXcluster membership from affecting AUTOGEN's calculation of VAXcluster-related parameter values. Define the NUM_NODES symbol in MODPARAMS.DAT to specify the number of nodes that are to run in the VAXcluster. AUTOGEN uses this value to set parameters that are affected by the number of VAXcluster nodes. For example, include the following line in MODPARAMS.DAT:

```
NUM_NODES = 30
```

Note

String values must be within quotation marks ("").

1.5.1.6. Defining the Number of Ethernet Adapters (VAX Only)

In a VAXcluster environment, define the NUM_ETHERADAPT symbol in MODPARAMS.DAT to specify the total number of Ethernet adapters in the VAXcluster system. For example, include the following line in MODPARAMS.DAT:

```
NUM_ETHERADAPT = 40
```

1.5.1.7. Presetting Parameter Values Before Adding Memory (VAX Only)

On VAX systems, if you plan to upgrade your system hardware by adding a large amount (512 MB or more) of memory, you might want to preset your system parameters to values appropriate for the

additional memory. Presetting your system parameters minimizes the possibility of memory upgrade problems caused by inappropriate parameter values.

How to Perform This Task

Perform the following steps:

1. Add a line to SYS$SYSTEM:MODPARAMS.DAT in the following format:

   ```
   MEMSIZE = total-number-of-pages-of-memory-after-upgrade
   ```

   For example:

   ```
   MEMSIZE = 2048 * 1024 ! (2048 page per MB *  1GB of memory)
   ```

2. Run AUTOGEN to the SETPARAMS phase.

3. Perform the hardware upgrade to add the additional memory.

4. Edit MODPARAMS.DAT to remove the line added in step 1.

1.5.1.8. Overriding Parameters Related to DECnet

To override AUTOGEN's observations regarding the presence (or absence) of DECnet, set the MODPARAMS.DAT parameter LOAD_DECNET_IMAGES to TRUE (or FALSE). Controlling the setting is useful for sites that have no synchronous network hardware but want to run asynchronous DECnet.

1.5.1.9. Values Set for NPAGEDYN and NPAGEVIR

For the benefit of OpenVMS VAX systems with limited physical memory, AUTOGEN logs a warning message in its report if NPAGEDYN exceeds 10 percent of physical memory or if NPAGEVIR exceeds 33 percent of physical memory.

AUTOGEN also limits its own calculated value for NPAGEDYN to 20 percent of physical memory, and limits NPAGEVIR to 50 percent of physical memory. These calculated values are adequate for most workstations and systems with 16 or fewer megabytes of physical memory. If your system requires a larger value, you can override the AUTOGEN calculated values by setting higher values in MODPARAMS.DAT.

1.5.2. Specifying a Minimum Required Age for Feedback (VAX Only)

On VAX systems, AUTOGEN feedback is useful only when a system has been running long enough to accurately reflect the system's normal work load. By default, AUTOGEN uses feedback if the data is older than 24 hours. On VAX systems, you can define the logical name AGEN $FEEDBACK_REQ_TIME to specify, in hours, a different minimum age required for feedback. AUTOGEN uses this value to determine whether the feedback is to be used.

For example, you might define the logical name as follows, to indicate that AUTOGEN should use feedback if it is older than 19 hours:

```
$ DEFINE/SYSTEM AGEN$FEEDBACK_REQ_TIME 19
```

To define this logical name each time the system starts up, add this command to SYLOGICALS.COM.

1.5.3. Including an External Parameter File in MODPARAMS.DAT

You can include external parameter files in MODPARAMS.DAT. For example, you might want to set a system parameter to the same value on all nodes in a VAXcluster or an OpenVMS Cluster environment; you might also want to specify node-specific values for other system parameters. You could specify the cluster-common values in a separate cluster-common file and include this cluster-common file in the MODPARAMS.DAT file on each system in the cluster.

To include a parameter file, place a command in the following format in MODPARAMS.DAT, or in any parameter file that is included in MODPARAMS.DAT:

```
AGEN$INCLUDE_PARAMS full-directory-spec:filename
```

Example

To include a cluster-common parameter file named CLUSTERPARAMS.DAT, create a common parameter file with the following name:

```
SYS$COMMON:[SYSEXE]CLUSTERPARAMS.DAT
```

Add the following line in the MODPARAMS.DAT file in the system-specific directory of each cluster:

```
AGEN$INCLUDE_PARAMS SYS$COMMON:[SYSEXE]CLUSTERPARAMS.DAT
```

1.5.4. Turning Off Logging of DCL Statements

The contents of MODPARAMS.DAT are evaluated as DCL statements; you can make assignments to symbols with names that are not system parameters (for example, scratch variables or conditional assignments based on other values). Traditionally, every such assignment is logged in AGEN $PARAMS.REPORT, sometimes creating a large file with many logging statements that do not interest users.

You can prefix any assignments that you prefer not to log in AGEN$PARAMS.REPORT with a dollar sign ($). When AUTOGEN encounters a MODPARAMS.DAT record beginning with a dollar sign, it does not check the list of known system parameters and does not log this record to AGEN $PARAMS.REPORT.

1.6. Automating AUTOGEN Reports

VSI recommends that you create a batch-oriented command procedure to automatically run AUTOGEN on a regular basis and send the resulting feedback reports to an appropriate Mail account. *Example 1.2, "Sample AUTOGEN Command Procedure "* provides a sample command procedure.

Note

This command procedure runs AUTOGEN only to recommend system parameter values and send you a report. It does not run AUTOGEN to change system parameters or reboot the system. If, after reviewing

the report, you decide to change system parameters, follow the instructions in *Section 1.6.1, "Changing Parameter Values After Reviewing AUTOGEN Reports"*.

The command procedure in *Example 1.2, "Sample AUTOGEN Command Procedure "* runs two passes of AUTOGEN. On the first pass, AUTOGEN runs during peak workload times to collect data on realistic system work loads. This pass does not degrade system performance. On the second pass, AUTOGEN runs during off-peak hours to interpret the data collected in the first stage.

The procedure sends the resulting report, contained in the file AGEN$PARAMS.REPORT, to the SYSTEM account. Review this report on a regular basis to see whether the load on the system has changed.

Example 1.2, "Sample AUTOGEN Command Procedure " shows a sample command procedure. Use this procedure only as an example; create a similar command procedure as necessary to meet the needs of your configuration.

Example 1.2. Sample AUTOGEN Command Procedure

```
$ BEGIN$:   ! ++++++++++ AGEN_BATCH.COM ++++++++++
$  on warning then goto error$
$  on control_y then goto error$
$!
$! Setup process
$!
$! Set process information
$  set process/priv=all/name="AUTOGEN Batch"
$! Keep log files to a reasonable amount
$  purge/keep=5 AGEN_Batch.log
$  time = f$time()    ! Fetch current time
$  hour = f$integer(f$cvtime(time, , "hour")) ! Get hour
$  today = f$cvtime(time, , "WEEKDAY")  ! Get Day of the week
$  if f$integer(f$cvtime(time, , "minute")) .ge. 30 then hour = hour + 1
$!
$! Start of working day...
$!
$ 1AM$:
$  if hour .le. 2
$     then
$     next_time = "today+0-14"
$     gosub submit$    ! Resubmit yourself
$     set noon
$!

$!      Run AUTOGEN to TESTFILES using the parameter values collected
 earlier

$!      in the day (i.e., yesterday at 2:00pm)
$       if today .eqs. "Tuesday" .OR. today .eqs. "Thursday" .OR. -
    today .eqs. "Saturday"
$          then
$      @sys$update:autogen GETDATA TESTFILES feedback (2)
$      mail/sub="AUTOGEN Feedback Report for system-name" -
  sys$system:agen$params.report system (3)
$      ! Clean up
$            purge/keep=7 sys$system:agen$feedback.report (4)
$            purge/keep=7 sys$system:agen$feedback.dat
```

```
$            purge/keep=7 sys$system:params.dat
$            purge/keep=7 sys$system:autogen.par
$            purge/keep=7 sys$system:setparams.dat
$            purge/keep=7 sys$system:agen$addhistory.tmp
$            purge/keep=7 sys$system:agen$addhistory.dat
$        endif
$     goto end$
$     endif
$!
$ 2PM$:
$  if hour .le. 15
$     then
$     next_time = "today+0-17"
$     gosub submit$
$     if today .eqs. "Monday" .OR. today .eqs. "Wednesday" .OR. -
  today .eqs. "Friday"
$        then
$          @sys$update:autogen SAVPARAMS SAVPARAMS feedback (1)
$        endif
$     goto end$
$     endif
$!
$ 5PM$:
$  if hour .le. 18
$     then
$     next_time = "tomorrow+0-1"
$     gosub submit$
$     endif
$!
$! End of working day...
$!
$ END$:     ! ---------- BATCH.COM ----------
$  exit
$!++
$! Subroutines
$!--
$!
$ SUBMIT$:
$  submit/name="AGEN_Batch"/restart/noprint - (5)
   /log=AGEN_batch.log -
   /queue=sys$batch/after="''next_time'" sys$system:AGEN_batch.com
$  return
$!++
$! Error handler
$!--
$ ERROR$:
$  mail/sub="AGEN_BATCH.COM - Procedure failed." _nl: system
$  goto end$
```

The commands in this procedure perform the following tasks:

1. Executes the first pass of AUTOGEN during peak workload times to collect data on realistic work loads. This command runs a very fast image so it does not degrade system response.

2. Executes the second pass of AUTOGEN during off-peak hours to interpret the data collected in the first pass.

3. Mails the resulting report file named AGEN$PARAMS.REPORT to the SYSTEM account.

4. Cleans up the files created.

5. Resubmits the command procedure.

1.6.1. Changing Parameter Values After Reviewing AUTOGEN Reports

If the command procedure report described in *Section 1.6, "Automating AUTOGEN Reports"* shows AUTOGEN's calculations are different from the current values, correct the tuning by executing AUTOGEN with one of the two following commands:

- If the system can be shut down and rebooted immediately, execute the following command:

  ```
  $ @SYS$UPDATE:AUTOGEN GETDATA REBOOT FEEDBACK
  ```

- If the system cannot be shut down and rebooted immediately, execute the following command to reset the system parameters:

  ```
  $ @SYS$UPDATE:AUTOGEN GETDATA SETPARAMS FEEDBACK
  ```

  The new parameters will take effect the next time the system boots.

1.7. Managing System Parameters with SYSMAN

Note

VSI recommends that you use AUTOGEN to modify system parameters. For more information, see *Section 1.5, "Modifying System Parameters with AUTOGEN"*. To view system parameters for a group of nodes or change parameters temporarily, use the System Management utility (SYSMAN).

SYSMAN provides the ability to inspect and modify system parameters for an entire cluster or for any group of nodes, rather than just one system. The PARAMETERS commands available in SYSMAN duplicate the parameter functions of the System Generation utility (SYSGEN).

You can use SYSMAN to manage system parameters as follows:

Task	**For More Information**
Show parameter values	*Section 1.7.2, "Showing Parameter Values with SYSMAN"*
Modify current values in the parameter file	*Section 1.7.3, "Modifying a Parameter File with SYSMAN"*
Modify active values on a running system (Applies only to dynamic system parameters.)	*Section 1.7.4, "Modifying Active*

Task	For More Information
	Values with SYSMAN"

SYSMAN provides the commands and functions shown in *Table 1.3, "SYSMAN PARAMETERS Commands"*.

Table 1.3. SYSMAN PARAMETERS Commands

Command	Function
PARAMETERS SHOW	Displays parameter values. Requires the name of the parameter.
PARAMETERS USE	Reads a set of parameters from memory or disk into the work area for inspection or modification. Requires a file name or the additional parameters ACTIVE or CURRENT.
PARAMETERS SET	Changes parameter values only in the work area; more permanent modification requires the PARAMETERS WRITE command. Requires the name and value of the parameter.
PARAMETERS WRITE	Writes the content of the work area to memory or to disk. Requires a file name or the additional parameters ACTIVE or CURRENT.

For more information about the temporary work area, see the next section.

1.7.1. Understanding Parameter Values and SYSMAN

It helps to understand the different system parameter values explained in *Section 1.1.1, "Default, CURRENT, and ACTIVE Values"*. Briefly, **current values** are stored in the default parameter file on disk. **Active values** are stored in memory and are used while the system is running. In addition to these values, SYSMAN writes a temporary copy into its own work area on disk. *Figure 1.2, "SYSMAN Temporary, Active, and Current Parameter Values"* illustrates these different sets of values and how SYSMAN commands affect them. In this figure:

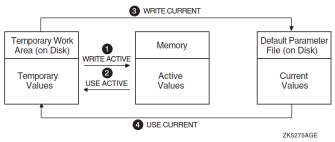
1. WRITE ACTIVE writes temporary parameter values to memory.

2. USE ACTIVE reads values from memory into the work area, where you can modify them.

3. WRITE CURRENT writes temporary parameter values to disk, where they become current values. They become active the next time the system boots.

4. USE CURRENT reads the current values from disk into the work area, where you can modify them.

Figure 1.2. SYSMAN Temporary, Active, and Current Parameter Values



During a typical session, you can display and change values in the following sequence:

1. Read values into SYSMAN's temporary work space with the USE command. USE ACTIVE reads in active values. USE CURRENT reads in current values.

2. Display the parameter values with the SHOW command.
3. Change a value with the SET command. You must use the WRITE command to activate the value.
4. Make the change effective with the WRITE command:
   - WRITE ACTIVE writes the value to the set of active values. (You can change an active value only if the parameter is a dynamic parameter.)
   - WRITE CURRENT writes the value to the set of current values.

For a list of all the system parameters, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

1.7.2. Showing Parameter Values with SYSMAN

Use the SYSMAN command PARAMETERS SHOW to display parameter values for all the nodes in a cluster.

Examples

1. The following example shows one method to display information about parameters. In this case, using the /LGI qualifier displays all login security control parameters. You can display many categories of parameters, such as /ACP, /ALL, and /SPECIAL. Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for a complete list of parameters and parameter categories.

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> PARAMETERS SHOW/LGI

Parameters in use: Active
Parameter Name    Current  Default   Min.   Max. Unit    Dynamic
--------------    -------  -------   -----  ---- ----    -------
LGI_BRK_TERM            0        1       0     1 Boolean    D
LGI_BRK_DISUSER         0        0       0     1 Boolean    D
LGI_PWD_TMO            30       30       0   255 Seconds    D
LGI_RETRY_LIM           3        3       0   255 Tries      D
LGI_RETRY_TMO          20       20       0   255 Seconds    D
LGI_BRK_LIM             5        5       0   255 Failures   D
LGI_BRK_TMO           300      300       0    -1 Seconds    D
LGI_HID_TIM           300      300       0    -1 Seconds    D
```

2. The following example invokes SYSMAN and specifies the environment to be the local cluster, which consists of NODE21 and NODE22. The example also displays the active value for the LGI_BRK_TMO parameter, which controls the number of seconds that a user, terminal, or node is permitted to attempt login. In this case, it is 600.

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> SET ENVIRONMENT/CLUSTER
%SYSMAN-I-ENV, Current command environment:
   Clusterwide on local cluster
   Username MORIN    will be used on nonlocal nodes
SYSMAN> PARAMETERS SHOW LGI_BRK_TMO

Node NODE21:   Parameters in use: ACTIVE
Parameter Name   Current   Default   Minimum    Maximum Unit  Dynamic
--------------   -------   -------   -------    ------- ----  -------
```

```
LGI_BRK_TMO           600       300         0         -1 Seconds     D

Node NODE22:   Parameters in use: ACTIVE
Parameter Name   Current   Default   Minimum    Maximum Unit  Dynamic
--------------   -------   -------   -------    ------- ----  -------
LGI_BRK_TMO           600       300         0         -1 Seconds     D
```

1.7.3. Modifying a Parameter File with SYSMAN

Use the SYSMAN command PARAMETERS WRITE to write system parameter values and the name of the site-independent startup command procedure to your choice of parameter file or the current system parameter file on disk.

The PARAMETERS WRITE CURRENT command sends a message to OPCOM to record the event, unless you have changed the system message format with the DCL command SET MESSAGE.

Note

The PARAMETERS WRITE CURRENT command writes *all* of the active or current parameter values – not just the one you may be working on – to disk.

Examples

1. The following example creates a new parameter specification file:

```
SYSMAN> PARAMETERS WRITE SYS$SYSTEM:NEWPARAM
```

2. When used with the PARAMETERS SET command, the PARAMETERS WRITE command modifies the current system parameter file on disk:

```
SYSMAN> PARAMETERS SET LGI_BRK_TMO 300
SYSMAN> PARAMETERS WRITE CURRENT
```

1.7.4. Modifying Active Values with SYSMAN

Using the SYSMAN commands PARAMETERS SET, PARAMETERS WRITE, and PARAMETERS USE enables you to modify active parameter values.

Modifying active values immediately affects dynamic parameters by changing their values in memory. Appendix C of the *VSI OpenVMS System Management Utilities Reference Manual* identifies dynamic parameters, as does the SYSMAN command PARAMETERS SHOW/DYNAMIC. Values for nondynamic parameters cannot be changed while the system is running.

Modifying active values does not affect current values in the system parameter file on disk, because the next time you boot the system, the values on disk are established as the active values.

If you set new active parameter values and you want to use the new values for subsequent boot operations, write the new values to the current parameter file with the PARAMETERS WRITE CURRENT command, as shown in the Examples section.

Caution

Parameter values modified with SYSMAN will be overridden by the AUTOGEN command procedure. To keep parameter modifications made with SYSMAN, edit the file SYS

$SYSTEM:MODPARAMS.DAT as explained in *Section 1.5.1, "Controlling AUTOGEN's Parameter Settings with MODPARAMS.DAT"* to specify the new parameter values.

Examples

1. The following example changes the LGI_BRK_TMO value to 300 in the work area, writes this change into memory as an active value, and displays the active value:

```
SYSMAN> PARAMETERS SET LGI_BRK_TMO 300

SYSMAN> PARAMETERS WRITE ACTIVE
SYSMAN> PARAMETERS SHOW LGI_BRK_TMO

Node NODE21:   Parameters in use: ACTIVE
Parameter Name   Current   Default   Minimum    Maximum Unit  Dynamic
--------------   -------   -------   -------    ------- ----  -------
LGI_BRK_TMO          300       300         0         -1 Seconds     D

Node NODE22:   Parameters in use: ACTIVE
Parameter Name   Current   Default   Minimum    Maximum Unit  Dynamic
--------------   -------   -------   -------    ------- ----  -------
LGI_BRK_TMO          300       300         0         -1 Seconds     D
```

2. The following example calls the current parameter values, including LGI_BRK_TMO, from disk to the work area, then displays LGI_BRK_TMO. In this example, the current value on disk is 600.

```
SYSMAN> PARAMETERS USE CURRENT
SYSMAN> PARAMETERS SHOW LGI_BRK_TMO

Node NODE21:   Parameters in use: CURRENT
Parameter Name   Current   Default   Minimum    Maximum Unit  Dynamic
--------------   -------   -------   -------    ------- ----  -------
LGI_BRK_TMO          600       300         0         -1 Seconds     D

Node NODE22:   Parameters in use: CURRENT
Parameter Name   Current   Default   Minimum    Maximum Unit  Dynamic
--------------   -------   -------   -------    ------- ----  -------
LGI_BRK_TMO          600       300         0         -1 Seconds     D
```

3. The next example writes the LGI_BRK_TMO value of 600 from the work area to memory, where it becomes the active value on the running system. Note that the command PARAMETER WRITE ACTIVE writes all the parameter values from the work area into memory, not just the value of LGI_BRK_TMO.

```
SYSMAN> PARAMETERS WRITE ACTIVE

SYSMAN> PARAMETERS USE ACTIVE

SYSMAN> PARAMETERS SHOW LGI_BRK_TMO

Node NODE21:   Parameters in use: ACTIVE
Parameter Name   Current   Default   Minimum    Maximum Unit  Dynamic
--------------   -------   -------   -------    ------- ----  -------
LGI_BRK_TMO          600       300         0         -1 Seconds     D

Node NODE22:   Parameters in use: ACTIVE
Parameter Name   Current   Default   Minimum    Maximum Unit  Dynamic
```

```
--------------   -------   -------   -------    ------- ----   -------
LGI_BRK_TMO          600       300         0         -1 Seconds     D
```

1.8. Managing System Parameters with SYSGEN

Note

VSI recommends that you use AUTOGEN to modify system parameters. For more information, see *Section 1.5, "Modifying System Parameters with AUTOGEN"*. If for some reason you cannot use AUTOGEN, VSI recommends that you use SYSMAN. For more information, see *Section 1.7, "Managing System Parameters with SYSMAN"*.

Although it is not the recommended method, you can also use the System Generation utility (SYSGEN) to manage system parameters as follows:

Task	For More Information
Show parameter values	*Section 1.8.2, "Showing Parameter Values with SYSGEN"*
Modify current values in the default parameter file	*Section 1.8.3, "Modifying the System Parameter File with SYSGEN"*
Modify active values on a running system [1]	*Section 1.8.4, "Modifying Active Values with SYSGEN"*
Create a new parameter file	*Section 1.8.5, "Creating a New Parameter File with SYSGEN"*

[1] Applies only to the dynamic system parameters.

SYSGEN provides the commands shown in *Table 1.4, "SYSGEN Commands Used with System Parameters"* for managing system parameters. Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for detailed descriptions of SYSGEN commands.

Table 1.4. SYSGEN Commands Used with System Parameters

Command	Function
SHOW	Displays parameter values.
USE	Reads a set of values from memory or disk into a temporary work area for inspection or modification.
SET	Changes parameter values only in the work area; more permanent modification requires the WRITE command.

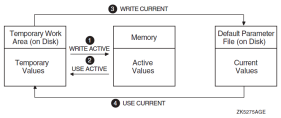
Command	Function
WRITE	Writes the content of the work area to memory or to disk.

For more information about the temporary work area, see the next section.

1.8.1. Understanding Parameter Values and SYSGEN

You should understand the different system parameter values explained in *Section 1.1.1, "Default, CURRENT, and ACTIVE Values"*. Briefly, CURRENT values are stored in the default parameter file on disk, and are used to establish initial parameter values when the system boots. ACTIVE values are stored in memory and are used while the system is running. In addition to these values, SYSGEN writes a temporary copy into its own work area on disk. *Figure 1.3, "SYSGEN Temporary, ACTIVE, and CURRENT Parameter Values"* illustrates these different sets of values and shows how SYSGEN commands affect them.

Figure 1.3. SYSGEN Temporary, ACTIVE, and CURRENT Parameter Values



In a typical session, you might display and change values in the following sequence:

1. Read values into SYSGEN's temporary work space with the USE command. USE ACTIVE reads in active values. USE CURRENT reads in current values.

2. Display the parameter values with the SHOW command.

3. Change a value with the SET command. (Note, however that the SET command only changes the value in SYSGEN's temporary work area.)

4. Make the change effective with the WRITE command:

   - WRITE ACTIVE writes the value to the set of active values in memory. (You can change an active value only if the parameter is a dynamic parameter.)

   - WRITE CURRENT writes the value to the set of current values on disk.

Note

SYSGEN modifications to the ACTIVE and CURRENT parameters may be monitored and reported by enabling the security auditing subsystem SYSGEN class. For example SET AUDIT/ENABLE=SYSGEN/ALARM or SET AUDIT/ENABLE=SYSGEN/AUDIT. In addition, changes to the CURRENT parameters will also send a message to the operator communication manager (OPCOM) to record the event in the operator log and notify any operator terminals and the operator console. Such a message will have a format similar to this:

```
%OPCOM, 15-APR-2019 16:04:06.30, message from user SYSTEM
%SYSGEN-I-WRITECUR, CURRENT system parameters modified by process ID
 00160030 into file ALPHAVMSSYS.PAR
```

For a list of all the system parameters, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

1.8.2. Showing Parameter Values with SYSGEN

To display values for system parameters, perform the following steps:

1. Invoke SYSGEN by entering the following command:

   ```
   $ RUN SYS$SYSTEM:SYSGEN
   ```

2. Enter the USE command to specify which values you want to display, as follows:

To Display	Enter
Active values	USE ACTIVE
Current values	USE CURRENT
Values from another parameter file	USE *file-spec* For *file-spec*, specify the parameter file from which you want to display values; for example, USE SYS$SYSTEM:ALTPARAMS.DAT

3. Enter a SHOW command in the following format:

   ```
   SHOW [/qualifier] [parameter-name]
   ```

   Specify qualifiers to display parameters grouped by type. For example:

To Display Values For	Enter
The WSMAX parameter	SHOW WSMAX
All dynamic parameters	SHOW/DYNAMIC
All parameters in the TTY category	SHOW/TTY
All parameters	SHOW/ALL

For more information about the SYSGEN SHOW command and qualifiers, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

Example

The following example uses SYSGEN to show the current values of all TTY system parameters:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> USE CURRENT
SYSGEN> SHOW/TTY

Parameters in use: Current❶
Parameter Name   Current    Default     Min.      Max. Unit       Dynamic
--------------   -------    -------  -------  -------  ----       -------
❷                   ❸          ❹        ❺        ❻       ❼
TTY_SCANDELTA   10000000   10000000  100000       -1  100Ns
TTY_DIALTYPE           0          0       0      255  Bit-Encode
TTY_SPEED             15         15       1       16  Special
TTY_RSPEED             0          0       0       16  Special
TTY_PARITY            24         24       0      255  Special
TTY_BUF               80         80       0    65535  Characters
TTY_DEFCHAR    402657952  402657952       0       -1  Bit-Encode
TTY_DEFCHAR2      135178       4098       0       -1  Bit-Encode
TTY_TYPAHDSZ          78         78       0       -1  Bytes
```

```
TTY_ALTYPAHD         2048         200         0      32767  Bytes
TTY_ALTALARM          750          64         0         -1  Bytes
TTY_DMASIZE            64          64         0         -1  Bytes       D ❽
TTY_CLASSNAME      "TTY"       "TTY"     "AA"       "ZZ"  Ascii
TTY_SILOTIME           8           8         0        255  Ms
TTY_TIMEOUT         3600         900         0         -1  Seconds     D
TTY_AUTOCHAR           7           7         0        255  Character   D
SYSGEN>
Parameters in use: Current
```

SYSGEN displays the following information:

❶ The values in use (in this example, current values)

❷ The name of the system parameter

❸ The value requested (in this example, the current value). The heading of this column is always "Current", regardless of whether it displays the current or active value of the parameter. In this context, "Current" refers to the value of this parameter *currently* in use, as specified by the USE command; it does not refer to the *current value* of the parameter stored on disk with the WRITE CURRENT command.

❹ The default value

❺ The minimum value

❻ The maximum value

❼ The unit of allocation

❽ A "D", if the system parameter is dynamic

1.8.3. Modifying the System Parameter File with SYSGEN

Caution

Parameter values modified with the System Generation utility (SYSGEN) will be overridden by the AUTOGEN command procedure. To keep parameter modifications made with SYSGEN, edit the file SYS$SYSTEM:MODPARAMS.DAT as explained in *Section 1.5.1, "Controlling AUTOGEN's Parameter Settings with MODPARAMS.DAT"* to specify the new parameter values.

Note

Although you can modify system parameter values with SYSGEN, VSI recommends that you use AUTOGEN. For more information, see *Section 1.5, "Modifying System Parameters with AUTOGEN"*.

If you cannot use AUTOGEN, VSI recommends that you use the System Management utility (SYSMAN) to modify system parameters. For more information, see *Section 1.7, "Managing System Parameters with SYSMAN"*.

Modifying the current values in the default system parameter file has no immediate effect on active values on a running system. However, during subsequent boot operations, the system is initialized with the new values.

Example

The following example modifies the TTY_TIMEOUT parameter value in the VAX system parameter file:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN SYSGEN
SYSGEN> USE CURRENT
SYSGEN> SET TTY_TIMEOUT 3600
SYSGEN> WRITE CURRENT
SYSGEN> EXIT
```

1.8.4. Modifying Active Values with SYSGEN

Caution

Parameter values modified with SYSGEN will be overridden by the AUTOGEN command procedure. To keep parameter modifications made with SYSGEN, edit the file SYS $SYSTEM:MODPARAMS.DAT as explained in *Section 1.5.1, "Controlling AUTOGEN's Parameter Settings with MODPARAMS.DAT"* to specify the new parameter value.

Note

Although you can modify system parameter values with SYSGEN, VSI recommends that you use AUTOGEN or the System Management utility (SYSMAN). For more information, see *Section 1.7, "Managing System Parameters with SYSMAN"*.

Modifying active values immediately affects dynamic parameters by changing their values in memory. *VSI OpenVMS System Management Utilities Reference Manual* identifies dynamic parameters (as does the SYSGEN command SHOW/DYNAMIC). You cannot change values for non dynamic parameters while the system is running.

Modifying active values does not affect the current values in the system parameter file on disk. The next time you boot the system, the old current values are established as the active values.

If you set new active parameter values (by entering WRITE ACTIVE) and you want to use the new values for subsequent boot operations, you must write the new values to the current parameter file on disk by entering the WRITE CURRENT command, as explained in *Section 1.8.3, "Modifying the System Parameter File with SYSGEN"*. If the parameters are not dynamic parameters, you must enter the WRITE CURRENT command and reboot the system.

Examples

1. The following example modifies the active value of the PFCDEFAULT parameter:

   ```
   $ SET DEFAULT SYS$SYSTEM
   $ RUN SYSGEN
   SYSGEN> SET PFCDEFAULT 127
   SYSGEN> WRITE ACTIVE
   SYSGEN> EXIT
   ```

2. The following example modifies the active value of the PFCDEFAULT parameter and also writes it to the Alpha system parameter file, so it will be used when the system reboots:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN SYSGEN
SYSGEN> SET PFCDEFAULT 127
SYSGEN> WRITE ACTIVE
SYSGEN> WRITE CURRENT
SYSGEN> EXIT
```

1.8.5. Creating a New Parameter File with SYSGEN

Creating a new parameter file has no effect on the running system. During a subsequent conversational boot operation, however, you can initialize the active system with the values of the new file.

How to Perform This Task

1. Invoke SYSGEN by entering the following commands:

   ```
   $ SET DEFAULT SYS$SYSTEM
   $ RUN SYSGEN
   ```

2. Enter a command in the following format to write a copy of a parameter file into SYSGEN's temporary workspace:

   ```
   USE file-spec
   ```

   where *file-spec* is the file specification for the parameter file to be used as a base. Modify the values in this file to create a new parameter file.

3. Enter commands in the following form to modify values as needed:

   ```
   SET parameter-name value
   ```

   where *parameter-name* specifies the name of the parameter to be changed, and *value* specifies the new value for the parameter.

4. Specify a command in the following format to write the values to a new parameter file:

   ```
   WRITE file-spec
   ```

   where *file-spec* is the file specification for the parameter file to be created.

5. Exit SYSGEN.

Caution

Parameter values modified with SYSGEN are overridden by the AUTOGEN command procedure. To keep parameter modifications made with SYSGEN, edit the file SYS$SYSTEM:MODPARAMS.DAT as explained in *Section 1.5.1, "Controlling AUTOGEN's Parameter Settings with MODPARAMS.DAT"* to specify the new parameter values.

Examples

1. The following example creates a new version of the parameter file PARAMS.PAR with a new value for the TTY_TIMEOUT parameter:

   ```
   $ SET DEFAULT SYS$SYSTEM
   ```

```
$ RUN SYSGEN
SYSGEN> USE SYS$MANAGER:PARAMS.PAR
SYSGEN> SET TTY_TIMEOUT 3600
SYSGEN> WRITE SYS$MANAGER:PARAMS.PAR
SYSGEN> EXIT
```

2. The following example creates a file named SYS$SYSTEM:OURSITE.PAR, using the PARAMS.PAR file as a base:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN SYSGEN
SYSGEN> USE SYS$MANAGER:PARAMS.PAR
SYSGEN> SET TTY_TIMEOUT 1000
SYSGEN> WRITE OURSITE.PAR
SYSGEN> EXIT
```

1.9. Modifying System Parameters with a Conversational Boot

Note

Although you can modify system parameters with a conversational boot, VSI recommends that you use AUTOGEN or the System Management utility (SYSMAN). For more information, see *Section 1.5, "Modifying System Parameters with AUTOGEN"* and *Section 1.7, "Managing System Parameters with SYSMAN"*.

Use a conversational boot only to change isolated system parameters *temporarily* or in an emergency. For example, during a system upgrade, you would use a conversational boot to modify STARTUP_P1 to use a minimum startup.

Remember that if you change a value and do not add the changed value to the AUTOGEN parameter file MODPARAMS.DAT, AUTOGEN will overwrite the value the next time AUTOGEN executes.

With a conversational boot operation, you can modify the active parameter values in the following ways before the system boots:

Task	For More Information
Modify active values for individual parameters	*VSI OpenVMS System Manager's Manual, Volume 1: Essentials*
Initialize active values using values stored in a parameter file other than the default parameter file	*VSI OpenVMS System Manager's Manual, Volume 1: Essentials*
Reinitialize active values using default values	*VSI OpenVMS System Manager's Manual, Volume 1: Essentials*

At the end of the conversational boot, the default system parameter file is modified to store the new active parameter values.

Caution

Parameter values modified with a conversational boot will be overridden by the AUTOGEN command procedure. To keep parameter modifications made with a conversational boot, edit the file SYS

$SYSTEM: MODPARAMS.DAT as explained in *Section 1.5.1, "Controlling AUTOGEN's Parameter Settings with MODPARAMS.DAT"* to specify the new parameter values.

1.10. Tuning BAP System Parameters

OpenVMS Alpha Version 7.1 and later contains system parameters that control the operation of bus-addressable pool (BAP).

The CIPCA, CIXCD, KFMSB, and Qlogic 1020ISP adapters are some of the adapters that use bus-addressable pool to improve performance. BAP is a non-paged dynamic, physical-address-filtered memory pool used to overcome I/O bus and 32-bit adapter physical addressing limits.

The following table lists the system parameters that control BAP operation along with their default values:

System Parameter	Default Value
NPAG_BAP_MIN	0
NPAG_BAP_MAX	0
NPAG_BAP_MIN_PA	0
NPAG_BAP_MAX_PA	-1

The default values of these parameters allow the system to boot with any configuration. When AUTOGEN is run on a configured system, it resets these parameters to values that should enhance performance for the current system configuration.

If the system fails to boot after an installation, upgrade, or configuration change, and displays a message that refers to incorrect BAP parameters, VSI recommends that you perform the following steps:

1. Reset the BAP parameters to the default values.
2. Reboot the system.
3. Allow the installation procedure to run AUTOGEN, or manually run AUTOGEN yourself.

A typical AUTOGEN with FEEDBACK command to set these parameters follows:

```
$ @SYS$UPDATE:AUTOGEN SAVPARAMS SETPARAMS FEEDBACK
```

Note

These parameters are critical. VSI recommends that you run AUTOGEN as described to ensure that they are set correctly.

If you prefer not to use this command because you want to adjust only the BAP parameters settings, use the following procedure:

1. Boot the system using the default BAP parameter values.
2. Manually run SYS$SYSTEM:AGEN$FEEDBACK.EXE:

   ```
   $ RUN SYS$SYSTEM:AGEN$FEEDBACK.EXE
   ```

3. Search SYS$SYSTEM:AGEN$FEEDBACK.DAT for the BAP_* system parameter values:

```
$ SEARCH SYS$SYSTEM:AGEN$FEEDBACK.DAT "BAP_"
```

4. Run SYSGEN to set the following system parameters with the BAP values you obtained in Step 3:

AGEN$FEEDBACK Data	System Parameter	Units
BAP_MIN	NPAG_BAP_MIN	bytes
BAP_MAX	NPAG_BAP_MAX	bytes
BAP_MIN_PA	NPAG_BAP_MIN_PA	Mbytes[1]
BAP_MAX_PA	NPAG_BAP_MAX_PA	Mbytes[b]

[1]On OpenVMS Alpha systems prior to Version 7.2, the value of this parameter is specified in bytes.

[b]On OpenVMS Alpha systems prior to Version 7.2, the value of this parameter is specified in bytes.

The BAP allocation amount (specified by BAP_MIN and BAP_MAX) depends on the adapter type, the number of adapters, and the version of the operating system. The physical address range (specified by BAP_MIN_PA and BAP_MAX_PA) depends on the adapter type and the way the Galaxy logical partitions, if any, are defined.

Note

If you manually set parameters NPAG_BAP_MIN_PA and NPAG_BAP_MAX_PA, be sure to specify the value for each parameter in the correct units (bytes or megabytes) for your operating system version.

Chapter 2. Managing Page, Swap, and Dump Files

Page, swap, and dump files are created by default. However, you should understand these files. In addition, you might want to change them to meet the needs of your site.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Displaying information about page and swap files	*Section 2.3, "Displaying Information About Page and Swap Files"*
Calculating appropriate sizes for files	*Section 2.4, "Manually Calculating Appropriate Sizes for Dump, Page, and Swap Files"*
Minimizing dump file size when disk space is insufficient	*Section 2.5, "Minimizing System Dump File Size When Disk Space Is Insufficient"*
Writing a dump file on a system disk with multiple paths or shadow set members	*Section 2.6, "Writing the System Dump File to the System Disk"*
Writing the dump file to a device other than the system disk	*Section 2.7, "Writing the System Dump File to an Alternate Disk"*
Using SDA to analyze the contents of a crash dump	*Section 2.8, "Using SDA to Analyze the Contents of a Crash Dump"*
Using SDA CLUE commands to obtain and analyze summary crash dump information[1]	*Section 2.8.1, "Using SDA CLUE Commands to Analyze Crash Dump Files (Alpha Only)"*
Using CLUE to obtain historical information about crash dumps[b]	*Section 2.9, "Using CLUE to*

Task	Section
	Obtain Historical Information About Crash Dumps(VAX Only)"
Saving the contents of the system dump file after a system failure	*Section 2.10, "Saving the Contents of the System Dump File After a System Failure"*
Copying dump files to tape or disk	*Section 2.11, "Copying System Dump Files to Tape or Disk"*
Freeing dump information from the page file	*Section 2.12, "Freeing Dump Information from the Page File"*
Installing page and swap files	*Section 2.13, "Installing Page and Swap Files"*
Removing page, swap, and dump files	*Section 2.14, "Removing Page, Swap, and Dump Files"*
Creating and modifying page, swap, and dump files	*Section 2.15, "Creating and Modifying Page, Swap, and Dump Files"*
Controlling access to process dumps	*Section 2.16.2, "Granting Access to Process Dumps (Alpha Only)"*

[1]Alpha and I64 specific
[b]VAX specific

This chapter explains the following concepts:

Concept	Section
Understanding dump files	*Section 2.1, "Understanding Dump Files"*
Understanding page and swap files	*Section 2.2, "Understanding Page and Swap Files"*

Concept	Section
Understanding the order of information in a selective system dump	*Section 2.5.1, "Understanding the Order of Information in a Selective System Dump"*
Understanding SDA CLUE[1]	*Section 2.8.1.1, "Understanding CLUE (Alpha Only)"*
Understanding CLUE[b]	*Section 2.9.1, "Understanding CLUE(VAX Only)"*
Understanding process dumps	*Section 2.16, "Understanding Process Dumps"*

[1]Alpha and I64 specific

[b]VAX specific

2.1. Understanding Dump Files

When the operating system detects an unrecoverable error or an inconsistency within itself that causes the system to fail, it writes the contents of the error log buffers, processor registers, and memory into the **system dump file**, overwriting its previous contents.

The contents of error log buffers are also written to the **error log dump file**. The error log dump file is provided so that the system can be updated on reboot to include error log entries that were created but not written at the time of a system crash.

System Dump File

When writing the system dump file, the system displays console messages and information about the error or inconsistency. The last message tells you that the dump file was successfully written.

Caution

Be sure to wait until you see the termination message before using the console terminal to halt the system. If you do not wait, your system might not save a complete system dump file.

Console messages and the system dump file are important sources of information in determining the cause of a system failure. Use the contents in the following ways:

- Use the System Dump Analyzer utility (SDA) to analyze the contents of the dump and determine the cause of a failure.
- On Alpha systems, use SDA CLUE commands to obtain and analyze summary dump file information.
- On VAX systems, use CLUE to obtain historical information from system dump files.

- Contact your VSI support representative, indicating that you have made a copy of the contents of the system dump.

The default system dump file, SYS$SPECIFIC:[SYSEXE]SYSDUMP.DMP, is created during installation. (You do not need a system dump file to run the operating system. However, you must have a system dump file to diagnose system crashes.) AUTOGEN automatically determines an appropriate size for the system dump file for your hardware configuration and system parameters. Refer to *Section 2.5, "Minimizing System Dump File Size When Disk Space Is Insufficient"* for information about minimizing system dump file size if disk space is insufficient.

For special configurations or varying work loads, you can change the size of the system dump file. For information, see *Section 2.15.1, "Using AUTOGEN (Recommended Method)"*. You can write the system dump file on a disk other than the system disk. This is referred to as **dump off system disk** (DOSD). For more information, see *Section 2.7, "Writing the System Dump File to an Alternate Disk"*.

Error Log Dump File

AUTOGEN creates the error log dump file during installation; its size depends on your configuration and system parameters. Error log dump files on VAX and Alpha systems have the following differences:

- On Alpha systems, the error log dump file is called SYS$ERRLOG.DMP; it is located on the system disk. When an operator initiates a shutdown, the system writes the contents of error log buffers to the error log dump file and not to the system dump file. Thus, the last system crash dump is not overwritten.

- On VAX systems, the error log dump file is called SYSDUMP.DMP. How the system handles this file depends on whether you are using dump off system disk (DOSD):

  - With DOSD, error logs are written to a stub error log dump file, SYSDUMP.DMP, on the system disk. Also, both the error logs and system memory are written to the SYSDUMP.DMP file on the DOSD disk.

  - Without DOSD, both the error log and system memory are written to the SYSDUMP.DMP file on the system disk.

  On VAX systems with or without DOSD, the last system crash dump is always overwritten when an operator initiates a shutdown.

2.1.1. Using the Page File to Store System Crash Dumps

The operating system uses the latest version of SYS$SYSTEM:SYSDUMP.DMP to store system crash dumps. If SYSDUMP.DMP does not exist in SYS$SYSTEM, the operating system uses the system **page file**, SYS$SYSTEM:PAGEFILE.SYS, overwriting the contents of that file.

If the SAVEDUMP system parameter is set, the crash dump is retained in PAGEFILE.SYS when the system is booted. If SAVEDUMP is clear, the system uses the page file for paging; any dump written to the page file is lost.

If you use SYS$SYSTEM:PAGEFILE.SYS to capture system crash dumps, you should later free the space occupied by the dump for use in system paging, with either of the following methods:

- Use the SDA COPY command to copy the dump from the page file to a different file.

- Use the SDA RELEASE command to delete the information from the page file.

For detailed instructions, see *Section 2.12, "Freeing Dump Information from the Page File"*.

Include the appropriate commands in the SYSTARTUP_VMS.COM startup command procedure to free dump information from the page file each time the system reboots.

Caution

Be careful when using the page file for selective dumps. Selective dumps use up all available space. If your page file is small, selective dump information might fill the entire page file, leaving no space for paging during system boot. This can cause the system to hang during reboot.

2.1.2. Understanding Types of System Dumps

The two types of system dumps are physical and selective. *Table 2.1, "Definitions of Physical and Selective System Dumps"* defines physical and selective system dumps. *Table 2.3, "Comparison of Physical and Selective System Dump Files"* compares the information available in physical and selective system dump files.

Table 2.1. Definitions of Physical and Selective System Dumps

Type	Description
Physical dump	Writes the entire contents of physical memory to the system dump file. To ensure a useful physical dump, the system dump file must be large enough to contain all of physical memory.
Selective dump	Stores those portions of memory most likely to be useful in crash dump analysis. A selective system dump is useful when disk space is not available to hold all of physical memory.

Requirements for Creating a Useful System Dump

The following requirements must be met for the operating system to write a useful system dump file:

- The system parameter DUMPBUG must be set to 1 (the default value).
- If the system parameter SAVEDUMP is set to 0 (the default) the file SYS$SPECIFIC:[SYSEXE]SYSDUMP.DMP must exist on the system disk.
- If the file SYS$SPECIFIC:[SYSEXE]SYSDUMP.DMP does not exist on the system disk, the page file must be used to store the dump. The system parameter SAVEDUMP must be set to 1 and the file SYS$SPECIFIC:[SYSEXE]PAGEFILE.SYS must exist on the system disk.
- If the file SYS$SPECIFIC:[SYSEXE]SYSDUMP.DMP does not exist and SYS$SPECIFIC:[SYSEXE]PAGEFILE.SYS is not available for system dumps, you must create the dump file on an alternate disk (see *Section 2.7, "Writing the System Dump File to an Alternate Disk"*).
- If sufficient disk space is not available to allow a system dump file that can hold all of memory, the system parameter DUMPSTYLE must be set to the appropriate value to store a selective system dump. For more information, refer to *Section 2.5, "Minimizing System Dump File Size When Disk Space Is Insufficient"* and the *VSI OpenVMS System Management Utilities Reference Manual*.
- The system dump file (or page file if the SAVEDUMP system parameter is set) must be large enough to hold all information that is to be written if the system fails.

If the system parameter DUMPBUG is set, AUTOGEN automatically sizes SYSDUMP.DMP if enough disk space is available.

If the system parameter SAVEDUMP is set, AUTOGEN performs no operations on the system dump file.

AUTOGEN sizes the page file only for paging use, regardless of whether the SAVEDUMP system parameter is set.

BACKUP Considerations

A system dump file has the NOBACKUP attribute; therefore, the Backup utility (BACKUP) does not copy the file unless you use the qualifier /IGNORE=NOBACKUP when invoking BACKUP. When you use the SDA COPY command to copy the system dump file to another file, the operating system does not automatically set the new file to NOBACKUP. If you want to set the NOBACKUP attribute on the copy, use the SET FILE command with the /NOBACKUP qualifier as described in the *VSI OpenVMS DCL Dictionary*.

Security Considerations

By default, SYS$SYSTEM:SYSDUMP.DMP is protected against world access. Because a system dump file can contain privileged information, you should keep this level of protection on system dump files. Similarly, when you copy system dump files using the System Dump Analyzer utility (SDA) as explained in *Section 2.10, "Saving the Contents of the System Dump File After a System Failure"* and *Section 2.12, "Freeing Dump Information from the Page File"*, be sure to protect the copy from world read access. For more information about file protection, refer to the *VSI OpenVMS Guide to System Security*.

2.2. Understanding Page and Swap Files

As part of memory management, the operating system makes efficient use of physical memory by moving information between physical memory and files stored on disk. The system does this in two ways: paging and swapping. *Table 2.2, "Paging and Swapping Terminology"* defines these and related terms.

Table 2.2. Paging and Swapping Terminology

Term	Definition
Paging	A memory management operation that provides the efficient use of physical memory allotted to a *process*. Paging moves infrequently used portions of a process workspace out of physical memory to a file. For more information about paging, refer to the *OpenVMS Performance Management Manual* manual.
Page file	The file to which the system writes paged portions of memory. The OpenVMS installation process creates a page file named SYS$SYSTEM:PAGEFILE.SYS. If necessary, you can use SYS $SYSTEM:PAGEFILE.SYS in place of the system crash dump file. For more information, see *Section 2.1.1, "Using the Page File to Store System Crash Dumps"*.
Swapping	A memory management operation that provides the efficient use of physical memory available for the *entire system*. Swapping moves the entire workspace of a less active process out of physical memory to a file. For

Term	Definition
	more information about swapping, refer to the *OpenVMS Performance Management Manual* manual.
Swap file	The file to which the system writes swapped portions of memory. The OpenVMS installation procedure creates a swap file named SYS$SYSTEM:SWAPFILE.SYS.
Primary page and swap files	The default page and swap files created during OpenVMS installation. These files are named SYS$SYSTEM:PAGEFILE.SYS and SYS$SYSTEM:SWAPFILE.SYS.
Secondary page and swap files	Additional page and swap files that you might create for performance or disk space reasons. If you kept the primary page and swap file on the system disk, the system uses the space in the secondary files for paging and swapping in addition to the space in the primary page and swap files. For information about creating secondary page and swap files, see *Section 2.15, "Creating and Modifying Page, Swap, and Dump Files"*.

Installing Files

Page and swap files must be installed before the system can use them. The system automatically installs the latest versions of SYS$SYSTEM:PAGEFILE.SYS and SWAPFILE.SYS during startup. If you create secondary page and swap files, you must make sure the system installs them during startup. For more information about installing page and swap files, see *Section 2.13, "Installing Page and Swap Files"*.

File Sizes and Locations

AUTOGEN automatically determines appropriate sizes for the files for your hardware configuration and system parameters. For special configurations or varying work loads, you might want to change the size of the page or swap file. For information, see *Section 2.15.1, "Using AUTOGEN (Recommended Method)"*.

If your system does not require the page file for storing system crash dumps, you can move it off the system disk. However, you should keep one page file on the system disk, if possible, so that you can boot the system if another disk holding the page files becomes unavailable. The swap file can also be moved off the system disk.

2.3. Displaying Information About Page and Swap Files

The DCL command SHOW MEMORY/FILES displays information about the page and swap files existing on your system, including file names, sizes, and the amount of space used. For example:

```
$ SHOW MEMORY/FILES
              System Memory Resources on 19-JAN-2017 13:35:26.58
 Swap File Usage (8KB pages):                    Index     Free      Size
   DISK$PAGE_DUMPS:[SYS0.SYSEXE]SWAPFILE.SYS;2
                                                     1     7992      8248
 Paging File Usage (8KB pages):                  Index     Free      Size
   DISK$PAGE_DUMPS:[SYS0.SYSEXE]PAGEFILE.SYS;1
                                                   254    13722    16496
   Total committed paging file usage:                                 4870
```

The total committed paging file usage is the number of pages in the system that require pagefile space for paging. It can be bigger than the total number of pagefile pages that are available because it is unlikely that all the required space will be used for paging at one time.

2.4. Manually Calculating Appropriate Sizes for Dump, Page, and Swap Files

When you install or upgrade the operating system, AUTOGEN automatically calculates appropriate sizes for your system; these sizes are based on your hardware configuration and your system parameters. However, you can manually calculate the sizes for these files. The following sections describe how to determine appropriate sizes for the system page, swap, and dump files.

2.4.1. Calculating System Dump File Size

Sufficient space in the system dump file is critical to saving a complete crash dump. The AUTOGEN command procedure calculates an appropriate size for your system dump file. However, if you want to manually calculate the system dump file size, use the following formula, which calculates the file size required to hold a physical dump.

For SYSDUMP.DMP

On VAX systems, use the following formula:

```
size-in-blocks(SYS$SYSTEM:SYSDUMP.DMP)
= size-in-pages(physical-memory)
+ number-of-error-log-buffers * blocks-per-buffer
+ 1
```

On Alpha systems, use the following formula:

```
size-in-blocks(SYS$SYSTEM:SYSDUMP.DMP)
= size-in-pages(physical-memory) * blocks-per-page
+ number-of-error-log-buffers * blocks-per-buffer
+ 10
```

where:

`size-in-pages`	Is the size of physical memory, in pages. Use the DCL command SHOW MEMORY to determine the total size of physical memory on your system.

`blocks-per-page`	Is the number of blocks per page of memory.
	
	On Alpha systems, calculate the number of blocks per page of memory by dividing the system's page size by 512 (the size of a block). Use the following commands:
	
	`$ PAGESIZE==F$GETSYI ("PAGE_SIZE")`
	`$ BLOCKSPERPAGE=PAGESIZE/512`
	`$ SHOW SYMBOL BLOCKSPERPAGE`

`number-of-error-log-buffers`	Is the value of the system parameter ERRORLOGBUFFERS. This parameter sets the

	number of error log buffers to permanently allocate in memory.
`blocks-per-buffer`	Is the value of the system parameter ERLBUFFERPAGES. This parameter sets the number of pagelets (blocks) of memory in each buffer.

A large memory system or a system with small disk capacity might not be able to supply enough disk space for a full memory dump. Under these circumstances, you should set the system parameter DUMPSTYLE to the appropriate value to indicate that the system is to dump only selective information. For more information, see *Section 2.5, "Minimizing System Dump File Size When Disk Space Is Insufficient"*.

For PAGEFILE.SYS

If SYS$SYSTEM:SYSDUMP.DMP does not exist, the system writes crash dumps to the primary page file SYS$SYSTEM:PAGEFILE.SYS. The AUTOGEN command procedure calculates an appropriate size for your page file. However, to manually calculate the minimum page file size required to hold crash dumps, use the following formula:

On VAX systems:

```
size-in-blocks(SYS$SYSTEM:PAGEFILE.SYS)
= size-in-pages(physical-memory)
+ number-of-error-log-buffers * blocks-per-buffer
+ 1
+ 1000
```

On Alpha systems:

```
size-in-blocks(SYS$SYSTEM:PAGEFILE.SYS)
= size-in-pages(physical-memory) * blocks-per-page
+ number-of-error-log-buffers * blocks-per-buffer
+10
+ value of the system parameter RSRVPAGCNT
```

where:

`size-in-pages`	Is the size of physical memory, in pages. Use the DCL command SHOW MEMORY to determine the total size of physical memory on your system.

`blocks-per-page`	Is the number of blocks per page of memory.
	
	On Alpha systems, calculate the number of blocks per page of memory by dividing the system's page size by 512 (the size of a block). Use the following commands:
	
	`$ PAGESIZE==F$GETSYI ("PAGE_SIZE")`
	`$ BLOCKSPERPAGE=PAGESIZE/512`
	`$ SHOW SYMBOL BLOCKSPERPAGE`

`number-of-error-log-buffers`	Is the value of the system parameter ERRORLOGBUFFERS. This parameter sets the

	number of error log buffers to permanently allocate in memory.
blocks-per-buffer	Is the value of the system parameter ERLBUFFERPAGES. This parameter sets the number of pagelets (blocks) of memory in each buffer.
RSRVPAGCNT	Is the value of the RSRVPAGCNT special system parameter.

Caution

This formula calculates only the minimum size requirement for saving a dump in the system's primary page file. For most systems, the page file must be larger than this to avoid hanging the system. For more information about calculating the page file size, see *Section 2.4.3, "Calculating Page File Size"*.

2.4.2. Calculating Error Log Dump File Size

These calculations differ on OpenVMS VAX and Alpha systems:

On Alpha Systems

On Alpha systems, the AUTOGEN command procedure calculates the appropriate size of your error log dump file. However, to calculate the size of the file manually, use the following formula, which calculates the file size required to hold all the error log buffers:

```
size-in-blocks(SYS$SYSTEM:SYS$ERRLOG.DMP)
= number-of-error-log-buffers * blocks-per-buffer
+ 2
```

where:

number-of-error-log-buffers	Is the value of the system parameter ERRORLOGBUFFERS. This parameter sets the number of error log buffers that are permanently allocated in memory.
blocks-per-buffer	Is the value of the system parameter ERLBUFFERPAGES. This parameter sets the number of pagelets (blocks) of memory in each buffer.

On VAX Systems

* maximum-number-of-processes

* maximum-number-of-processes

On VAX systems, the size of the error log dump file depends on your DOSD:

- Without DOSD, error logs are written to the dump file SYSDUMP.DMP on the system disk. See *Section 2.4.1, "Calculating System Dump File Size"* for sizing information.

- With DOSD:
  - Both error logs and system memory are written to the SYSDUMP.DMP file on the DOSD disk. See *Section 2.4.1, "Calculating System Dump File Size"* for sizing information.
  - In addition, error logs are written to the stub system dump file SYSDUMP.DMP on the system disk. Because this is a fixed-length file of 2048 blocks, no sizing calculations are required.

2.4.3. Calculating Page File Size

Sufficient page file space is critical to system performance. The AUTOGEN command procedure calculates an appropriate size for your page file space. The size calculated by AUTOGEN should be sufficient. However, to manually calculate the size for page file space, use one of the following formulas.

On VAX Systems

On VAX systems, use the following formula:

```
size-in-blocks (total for all page files on the system)
= size-of-average-process (in pages)
* maximum-number-of-processes
```

- The `size-of-average-process` is the value of the average virtual size of the process. Use the following command to find it:

  ```
  $ SHOW PROCESS/CONTINUOUS/ID=pid
  ```

  Specify this value in pages.

- The `maximum-number-of-processes` is the value of the MAXPROCESSCNT system parameter.

If the result of the formula is less than VIRTUALPAGECNT, use the value of VIRTUALPAGECNT instead.

To determine a system's virtual page count, enter the following command:

```
$ WRITE SYS$OUTPUT F$GETSYI ("VIRTUALPAGECNT")
```

On Alpha Systems

On Alpha systems, no simple formula can be given because of the wide variation possible in memory size and usage. You can use the following formula for systems up to 512MB:

```
size-in-blocks (total for all page files on the system)
= size-of-average-process (in pages)
* blocks-per-page
* maximum-number-of-processes
```

- The `size-of-average-process` is the value of the average virtual size of the process. Use the following command to find it:

  ```
  $ SHOW PROCESS/CONTINUOUS/ID=pid
  ```

  Specify this value in pages.

- Calculate `blocks-per-page` by dividing the system page size by 512 (pagelet size). For example, a system with a page size of 8192 has 16 pagelets per page.

  To determine a system's page size, enter the following command:

  ```
  $ WRITE SYS$OUTPUT F$GETSYI ("PAGE_SIZE")
  ```

- The `maximum-number-of-processes` is the value of the MAXPROCESSCNT system parameter.

For systems over 512MB, follow the steps described below for monitoring page file usage (*Section 2.4.3.2, "Monitoring Page File Usage"*), and make adjustments as necessary.

2.4.3.1. Representing Page File Size

The page file size you calculate can be represented in one of the following ways:

- In the primary page file only
- Distributed across primary and secondary page files
- If you have removed the primary page file in SYS$SYSTEM, distributed across a number of secondary page files

2.4.3.2. Monitoring Page File Usage

Once you determine an initial size for your page file or files (either with AUTOGEN or manually), monitor page file usage by executing AUTOGEN with the following command:

```
$ @SYS$UPDATE:AUTOGEN SAVPARAMS TESTFILES FEEDBACK
```

With this command, AUTOGEN writes page file usage and size recommendations to the feedback report AGEN$PARAMS.REPORT. (For more information about AUTOGEN and the feedback report, see *Section 1.4, "Understanding the AUTOGEN Command Procedure"* and *Section 1.4.2, "Feedback Report (AGEN$PARAMS.REPORT)"*.) The DCL command SHOW MEMORY/FILES also displays file usage, as explained in *Section 2.3, "Displaying Information About Page and Swap Files"*.

Keep page file usage less than half the size of the page file or files. If a paging file starts to fill to the point where system performance is being affected, a message is printed on the console terminal. If this happens, increase the size of your page file or files or install additional files.

Note

Your system resources and work load affect the required size of your page file. You should be familiar with your system resources and work load. For more information, refer to the *OpenVMS Performance Management Manual*.

2.4.3.3. Limiting Page File Space

Limit the amount of page file space consumed by user programs by using the /PGFLQUOTA qualifier of the AUTHORIZE commands ADD and MODIFY. (Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for more information.) Do not reduce the value of /PGFLQUOTA below 1024. Size requirements of the page file vary widely, depending on user applications.

2.4.4. Calculating Swap File Size

Sufficient swap file space is critical to system performance. The AUTOGEN command procedure calculates an appropriate size for your swap file space. To manually calculate the size for swap file space, use the following formula:

```
 size-in-blocks (total for all swap files on the system)
 = Maximum-number-of-processes * Average-working-set-quota-of-processes-on-
system
 = maximum-number-of-processes * average-working-set-quota-of-processes-on-
system
```

where:

`maximum-number-of-processes`	Is the value of the MAXPROCESSCNT system parameter.
average-working-set-quota-of-processes-on-system	Is the average value of the WSQUOTA limit for processes running on the system. On VAX systems, specify the value in pages. On Alpha systems, specify the value in pagelets.

2.4.4.1. Representing Swap File Size

The size you calculate can be represented in any of the following ways:

- In the primary swap file only
- Distributed across primary and secondary swap files
- If you have removed the primary swap file in SYS$SYSTEM, distributed across a number of secondary swap files

2.4.4.2. Monitoring Swap File Usage

Once you have determined an appropriate size for swap file space (either manually or with AUTOGEN), monitor swap file usage with the DCL command SHOW MEMORY/FILES as explained in *Section 2.3, "Displaying Information About Page and Swap Files"*. Keep at least one-third of the swap file space unused; otherwise, system performance can be severely affected.

Note

Your system resources and work load affect the required size of your swap file. You should be familiar with your system resources and work load. For more information, refer to the *OpenVMS Performance Management Manual*.

2.5. Minimizing System Dump File Size When Disk Space Is Insufficient

In certain system configurations, it might be impossible to preserve the entire contents of memory in a disk file. For instance, a large memory system might not be able to supply enough disk space for a

full memory dump. If your system attempts to save all of memory but the dump file is too small to accommodate the entire dump, the System Dump Analyzer utility (SDA) might not be able to analyze the dump.

On VAX systems, insufficient dump space would also prevent the Crash Log Utility Extractor (CLUE) from being able to analyze the dump.

Options for Minimizing System Dump File Size

Use one of the following options to minimize the size of the system dump file when disk space is insufficient:

- Selective dumps

  To preserve those portions of memory that contain information most useful in determining the causes of system failures, you can use selective system dumps. *Table 2.1, "Definitions of Physical and Selective System Dumps"* defines physical and selective dumps.

  Table 2.3, "Comparison of Physical and Selective System Dump Files" compares the information available in physical and selective system dump files.

  Table 2.3. Comparison of Physical and Selective System Dump Files

Type	Available Information	Unavailable Information
Physical dump (also known as full dump)	Complete contents of physical memory in use, stored in order of increasing physical address and error log buffers.	Contents of paged-out memory at the time of the crash.
Selective dump	System page table, global page table, system space memory, error log buffers, and process and control regions (plus global pages) for all saved processes.	Contents of paged-out memory at the time of the crash, process and control regions of unsaved processes, and memory not mapped by a page table.

  To direct your system to save selective system dumps, set bit 0 of the system parameter DUMPSTYLE. For information about how to change system parameter values, see *VSI OpenVMS System Management Utilities Reference Manual*. For information about how to change system parameter values, see *Section 1.5, "Modifying System Parameters with AUTOGEN"*.

- Compressed dumps

  On Alpha systems, if you set bit 3 of the DUMPSTYLE system parameter, OpenVMS writes the physical or selective system dump in a compressed format. (The exact amount of the compression depends on system use, but the typical compressed dump is about 60 percent of its original size.) If you use compressed dumps, AUTOGEN sizes the system dump file at 2/3 of its uncompressed size.

- Dump off system disk (DOSD)

  If you set bit 2 of the DUMPSTYLE system parameter and meet all the other requirements, OpenVMS writes the system dump to a disk other than the system disk. For details, see *Section 2.7, "Writing the System Dump File to an Alternate Disk"*.

Section 2.5.1, "Understanding the Order of Information in a Selective System Dump" discusses the order in which information is written to a selective system dump on Alpha and VAX systems. *Section 2.5.2,*

"Fine-Tuning the Order That Processes Are Written in Selective System Dumps(Alpha Only)" discusses how this order can be fine-tuned on Alpha systems.

2.5.1. Understanding the Order of Information in a Selective System Dump

The following lists show the order in which information is written to selective dumps on VAX and Alpha systems.

On VAX systems, information is written to selective dumps in the following order:

1. System page table (SPT)
2. System space (including process page tables, page frame number (PFN) database, and global page table (GPT))
3. Global pages that appear in the working set of any process
4. Processes resident at the time of the crash:
    a. Current process on crash CPU
    b. Predefined processes (hardcoded into BUGCHECK)
    c. Current processes on other CPUs
    d. Other processes resident at the time of the crash, in order by process index

On Alpha systems, information is written to selective system dumps in the following order:

1. Page table (PT) space for shared addresses (S0/S1/S2)
2. S0/S1 space
3. S2 space
4. Any system space pages (P1, S0/S1, S2) that have been replicated for performance reasons, where the contents of the replicated page is different from the original
5. Memory map pages for Galaxy shared memory regions, if appropriate
6. Key processes:
    a. Current process on crash CPU
    b. Swapper
    c. Current processes on CPUs that have failed to record their crash state
    d. Current processes on other CPUs
    e. Site-specific priority processes (see next section)
    f. VSI-defined priority processes (hardcoded into BUGCHECK):

        MSCPmount

AUDIT_SERVER
NETACP
NET$ACP
REMACP
LES$ACP

7. Any processes in a resource or miscellaneous wait state (for example, RWAST)

8. Key global pages (those that appear in the working set of any key process)

9. Other processes (the non-key processes) resident at the time of the crash, in order by process index

10. Remaining global pages that appear in the working set of any non-key process

Note that on Alpha systems, processes are dumped in two stages: the page tables for the process first, and then the body of the process.

Usage Notes on VAX and Alpha Systems

On both VAX and Alpha platforms, no process is dumped twice. For example, on Alpha systems, if the current process is the Swapper, it is dumped only once.

Similarly, on Alpha systems, no global page is dumped twice. Therefore, if a page in the working set of a key process is dumped in the "Key global pages" section, it is not dumped again later just because it is also in the working set of a non-key process.

2.5.2. Fine-Tuning the Order That Processes Are Written in Selective System Dumps (Alpha Only)

On Alpha systems, a set of processes, known as **key processes**, are dumped immediately following PT, S0/S1, and S2, including transition pages that link back to the process. The system manager can designate additional processes to be treated as key processes. These processes have priority over other processes in a dump, thus ensuring that the selected processes are successfully written when the dump file is too small to contain all processes.

How to Perform This Task

To designate the order of processes in a dump, follow these steps:

1. Copy the file SYS$SYSTEM:SYS$DUMP_PRIORITY.TEMPLATE to SYS$SYSTEM:SYS $DUMP_PRIORITY.DAT.

2. Following the instructions in the file, edit the .DAT file to contain a list of the processes to be dumped early in the dump.

3. Run the image SYS$SYSTEM:SYS$SET_DUMP_PRIORITY.EXE. Note that the image is automatically run during system startup if the data file SYS$SYSTEM:SYS $DUMP_PRIORITY.DAT exists.

You can edit the data file and run the image at any time the system is running. Therefore, if a process is hung, the system manager can designate the process as a priority process and then force a crash.

2.6. Writing the System Dump File to the System Disk

If you have more than one path to the system disk, or the system disk is a shadow set with multiple members, you must take additional steps to ensure that a system dump can be written to the system disk.

2.6.1. System Dump to System Disk on Alpha

If there is more than one path to the system disk, the console environment variable DUMP_DEV must describe all paths to the system disk. This ensures that if the original boot path becomes unavailable because of failover, the system can still locate the system disk and write the system dump to it.

If the system disk shadow set has multiple members, the console environment variable DUMP_DEV must describe all paths to all members of the shadow set. This ensures that if the master member changes, the system can still locate the master member and write the system dump to it.

If you do not define DUMP_DEV, the system can write a system dump only to the physical disk used at boot time using only the same physical path used at boot time. For instructions on setting DUMP_DEV, see *Section 2.7.1, "DOSD Requirements on Alpha and I64 Systems"*.

Certain configurations (for example, those using Fibre Channel disks) may contain more combinations of paths to the system disk than can be listed in DUMP_DEV. In that case, VSI recommends that you include in DUMP_DEV all paths to what is usually the master member of the shadow set, because shadow set membership changes occur less often than path changes.

You can write the system dump to an alternate disk (see *Section 2.7.1, "DOSD Requirements on Alpha and I64 Systems"*), but when doing so you must still define a path to the system disk for writing error logs. Therefore, DUMP_DEV should contain all paths to the system disk in addition to the paths to the alternate dump disk.

If there are more paths than DUMP_DEV can contain, VSI recommends that you define all paths to the dump disk and as many paths as possible (but at least one) to the system disk. Note that the system disk paths must come last in the list.

2.6.2. System Dump to System Disk on VAX

To ensure that the system can locate the system disk and write the system dump to it when there is more than one path to the system disk, or when the system disk shadow set has multiple members, you must follow the platform-specific instructions regarding booting. On some VAX systems, you must set appropriate register values; on other VAX systems, you must set specific environment variables. See the upgrade and installation supplement for your VAX system for details.

Note that if the system has multiple CI star couplers, the shadow set members must all be connected through the same star coupler.

2.7. Writing the System Dump File to an Alternate Disk

You can write the system dump file to a device other than the system disk (DOSD) on OpenVMS systems. This is especially useful in large-memory systems and in clusters with common system

disks where sufficient disk space is not always available on one disk to support customer dump file requirements.

Requirements for DOSD are somewhat different on VAX and Alpha systems. On both systems, however, you must correctly enable the DUMPSTYLE system parameter to enable the bugcheck code to write the system dump file to an alternate device.

The following sections describe the requirements for DOSD on Alpha, I64, and VAX systems.

2.7.1. DOSD Requirements on Alpha and I64 Systems

On Alpha and I64 systems, DOSD has the following requirements:

- The dump device directory structure must resemble the current system disk structure. The [SYS *n*.SYSEXE]SYSDUMP.DMP file will reside there, with the same boot time system root.

  Use AUTOGEN to create this file. In the MODPARAMS.DAT file, the following symbol prompts AUTOGEN to create the file:

  ```
  DUMPFILE_DEVICE = "$nnn$ddcuuuu"
  ```

  You can enter a list of devices.

- The dump disk must have an ODS-2 or ODS-5 file structure.
- The dump device cannot be part of a volume set.
- Although not a requirement, VSI recommends that you mount the dump device during system startup. If the dump device is mounted, it can be accessed by CLUE and AUTOGEN and for the analysis of crash dumps. For best results, include the MOUNT command in SYPAGSWPFILES.COM.
- For the Crash Log Utility Extractor (CLUE) to support DOSD, you must define the logical name CLUE$DOSD_DEVICE to point to the dump file to be analyzed after a system crash. For instructions, refer to *Section 2.8.1, "Using SDA CLUE Commands to Analyze Crash Dump Files (Alpha Only)"*.
- The dump device cannot be part of a shadow set unless it is also the system device and the master member of the shadow set.
- Use the following format to specify the dump device environment variable DUMP_DEV at the console prompt:

  ```
  >>> SET DUMP_DEV device-name[...]
  ```

  Note

  On DEC 3000 series systems, the following restrictions on the use of the DUMP_DEV environment variable exist:

  - This variable is not preserved across system power failures because DEC 3000 series systems do not have enough nonvolatile RAM to save the contents of the file. You must reset the DUMP_DEV variable after a power failure. (DUMP_DEV is preserved across all other types of restarts and bootstraps, however.)
  - You cannot clear DUMP_DEV (except by power-cycling the system).

- You must use console firmware Version 6.0 or greater because earlier versions do not provide support for DUMP_DEV.

On some CPU types, you can enter only one device; on other CPU types, you can enter a list of devices. The list can include various alternate paths to the system disk and the dump disk.

- On I64 systems, use the OpenVMS I64 Boot Manager (BOOT_OPTIONS.COM) utility to specify the dump device environment variable DUMP_DEV at the OpenVMS DCL prompt:

```
$ @SYS$MANAGER:BOOT_OPTIONS
```

Alternatively, you can use the EFI for OpenVMS (I64 only) VMS_SET utility to specify the dump device environment variable DUMP_DEV at the EFI console prompt:

```
Shell> FSn:\EFI\VMS\VMS_SET DUMP_DEV device-name[,...]
(where: FSn: is any bootable FAT file-structured partition that includes
 the VMS_SET
utility)
```

Refer to the next section.

- By specifying an alternate path with DUMP_DEV, the disk can fail over to the alternate path when the system is running. If the system crashes subsequently, the bugcheck code can use the alternate path by referring to the contents of DUMP_DEV.
- When you enter a list of devices, however, paths to the system disk must come last.

Designating the Dump Device on Alpha Systems

To designate the dump device with the DUMP_DEV environment variable, and enable the DUMPSTYLE system parameter, follow these steps:

1. Display the value of BOOTDEF_DEV; for example:

```
>>> SHOW BOOTDEF_DEV

BOOTDEF_DEV             dub204.7.0.4.3,dua204.4.0.2.3
```

2. Display the devices on the system as follows:

```
>>> SHOW DEVICES

Resetting IO subsystem...

dua204.4.0.2.3     $4$DUA204 (RED70A)        RA72
dua206.4.0.2.3     $4$DUA206 (RED70A)        RA72
dua208.4.0.2.3     $4$DUA208 (RED70A)        RA72

polling for units on cixcd1, slot 4, xmi0...

dub204.7.0.4.3     $4$DUA204 (GRN70A)        RA72
dub206.7.0.4.3     $4$DUA206 (GRN70A)        RA72
dub208.7.0.4.3     $4$DUA208 (GRN70A)        RA72
>>>
```

In this example:

- DUA204 is the system disk device.
- DUA208 is the DOSD device.

3. To provide two paths to the system disk, with the dump disk as DUA208 (also with two paths), set DUMP_DEV as follows:

```
>>> SET DUMP_DEV
 dua208.4.0.2.3,dub208.7.0.4.3,dub204.7.0.4.3,dua204.4.0.2.3
```

In this example, dua208.4.0.2.3 and dub208.7.0.4.3 are paths to the dump device; dub204.7.0.4.3 and dua204.4.0.2.3 are paths to the boot device.

Note

The system chooses the first valid device that it finds in the list as the dump device. Therefore, the dump disk path entries must appear before the system disk entries in the list.

---

4. Display all environment variables on the system by entering the SHOW * command; for example:

```
>>> SHOW *

auto_action             HALT
baud                    9600
boot_dev  dua204.4.0.2.3
boot_file
boot_osflags            0,0
boot_reset              ON
bootdef_dev             dub204.7.0.4.3,dua204.4.0.2.3
booted_dev  dua204.4.0.2.3
booted_file
booted_osflags          0,0
cpu                     0
cpu_enabled             ff
cpu_primary             ff
d_harderr               halt
d_report                summary
d_softerr               continue
dump_dev  dua208.4.0.2.3,dub208.4.0.4.3,dub204.7.0.4.3,dua204.4.0.2.3
enable_audit            ON
interleave              default
language                36
pal                     V5.48-3/O1.35-2
prompt                  >>>
stored_argc             2
stored_argv0            B
stored_argv1            dua204.4.0.2.3
system_variant          0
version                 T4.3-4740 Jun 14 2017 15:16:38
>>>
```

After you complete this section, refer to “Section,” which follows the next section.

Designating the Dump Device on I64 Systems

You can designate the dump device on I64 systems by using either of the following:

- The Boot Manager Utility, BOOT_OPTIONS.COM
- The EFI utilities command VMS_SET.

Note

VSI recommends that you use the use the OpenVMS I64 Boot Manager Utility. (Use of the utility is optional for other devices but mandatory for Fibre Channel devices.) For more information on this utility, refer to the *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

Using the Boot Manager Utility BOOT_OPTIONS.COM

To designate the dump device with the DUMP_DEV environment variable from the OpenVMS DCL prompt, follow these steps:

1. At the DCL prompt, enter the following command to invoke the OpenVMS I64 Boot Manager utility:

   ```
   $ @SYS$MANAGER:BOOT_OPTIONS.COM
   ```

2. The utility starts by displaying the main menu. Set the utility to operate on the Dump Device Options list by entering D at the prompt:

   ```
   Enter your choice: D
   ```

   Sample output from adding an entry to the Dump Options list follows:

   ```
   OpenVMS I64 Boot Manager Dump Options List Management Utilities
    (1) ADD an entry to the Dump Options list
    (2) DISPLAY the Dump Options list
    (3) REMOVE an entry from the Dump Options list
    (4) MOVE the position of an entry in the Dump Options list
    (B) Set to operate on the Boot Device Options list
    (D) Set to operate on the Dump Device Options list
    (G) Set to operate on the Debug Device Options list
    (E) EXIT from the Boot Manager utility
   You can also enter Ctrl-Y at any time to abort this utility.
   ```

   #### Note

   While using this utility, you can change a response by typing the up-arrow (^) as many times as you need to. To end the program and return to the DCL prompt, enter Ctrl/Y.

3. To designate the dump device on the DUMP_DEV environment variable, option, enter 1 at following prompt:

   ```
   Enter your choice: 1
   ```

4. The utility prompts you for the device name. Enter the device name of the DOSD device, as in the following example, in which the dump device is multipath fibre device DGA1:

   ```
   Enter the device name (Enter "?" for a list of devices): $1$DGA1
    efi$bcfg: $1$DGA1 (VMS_DUMP_DEV_01) Option successfully added
    efi$bcfg: $1$DGA1 (VMS_DUMP_DEV_02) Option successfully added
   ```

```
efi$bcfg: $1$DGA1 (VMS_DUMP_DEV_03) Option successfully added
```

5. Repeat steps 3 to 4 to add additional DUMP_DEV devices.

6. When you have successfully added all DUMP_DEV options, exit from the utility by entering E at the following prompt:

   ```
   Enter your choice: E
   ```

7. Reboot the system and refer to the *the section called "Enabling DOSD on Alpha and I64 Systems".*

Using the EFI Utilities for OpenVMS VMS_SET Command

You can use the EFI utilities for OpenVMS (I64 only) to specify the dump device environment variable DUMP_DEV at the EFI console prompt:

1. Display the devices on the system as follows:

   ```
   Shell> FS0:\EFI\VMS\VMS_SHOW DEVICE
    VMS: EIA0 0-30-6E-39-F7-A5
    EFI:Acpi(000222F0,0)/Pci(3|0)/Mac(00306E39F7A5
    VMS: DKA0 HP 18.2GATLAS10K3_18_SCAHP05
    EFI: Acpi(000222F0,100)/Pci(1|0)/Scsi(Pun0,Lun0)
    VMS: DKB400 HP 18.2GST318406LC HP05
    EFI: fs2: Acpi(000222F0,100)/Pci(1|1)/Scsi(Pun4,Lun0
    VMS: DKB200 HP 18.2GST318406LC HP05
    EFI: fs1: Acpi(000222F0,100)/Pci(1|1)/Scsi(Pun2,Lun0)
    VMS: DKB0 HP 18.2GATLAS10K3_18_SCAHP05
    EFI: fs0: Acpi(000222F0,100)/Pci(1|1)/Scsi(Pun0,Lun0)
    VMS: EWA0 0-30-6E-39-77-3
    EFI: Acpi(000222F0,100)/Pci(2|0)/Mac(00306E39773D
   ```

   In this example:

   - DKB0 and DKB200 are members of the system disk shadow set
   - DKA0 is the DOSD device.

2. Set DUMP_DEV as follows:

   ```
   Shell> FS0:\EFI\VMS\VMS_SET DKA0, DKB0, DKB200VMS: DKA0 HP
   18.2GATLAS10K3_18_SCAHP05EFI:
   Acpi(000222F0,100)/Pci(1|0)/Scsi(Pun0,Lun0)
   VMS: DKB0 HP 18.2GATLAS10K3_18_SCAHP05EFI: fs0:
   Acpi(000222F0,100)/Pci(1|1)/Scsi(Pun0,Lun0)
   VMS: DKB200 HP 18.2GST318406LC HP05EFI: fs1:
   Acpi(000222F0,100)/Pci(1|1)/Scsi(Pun2,Lun0)
   ```

   After you complete this section, refer to the next section.

Enabling DOSD on Alpha and I64 Systems

Finally, enable the DOSD bit of the DUMPSTYLE system parameter by setting bit 2. For example, enter the value of 4 at the SYSBOOT> prompt to designate an uncompressed physical dump to an alternate disk with minimal console output:

```
SYSBOOT> SET DUMPSTYLE 4
```

The *VSI OpenVMS System Management Utilities Reference Manual* and online help contain details about the DUMPSTYLE system parameter.

Note

The error log dump file is always created on the system disk so that error log buffers can be restored when the system is rebooted. This file is not affected by setting the DUMPSTYLE system parameter or the DUMP_DEV environmental variable.

The system chooses the first valid device that it finds in the list as the dump device. Therefore, the dump disk path entries must appear before the system disk entries in the list.

On Alpha systems, the number of devices that can be included in DUMP_DEV is limited. This limit varies from platform to platform, and it also depends on the device configuration. Some platforms allow only a single device; others allow a list. Systems that allow a list of devices limit the length of the list to 256 bytes in the internal format that the console subsystem uses. This length provides space for 4 entries if SCSI or CI-based disks are used, and 8 to 9 entries if fibre-channel disks are used.

On I64 systems, up to 99 devices can be included in DUMP_DEV, regardless of platform or device configuration.

2.7.2. DOSD Requirements on VAX Systems

On VAX systems, DOSD has the following requirements:

- The system must be connected directly to, and must boot from, CI controllers.
- The dump device must physically connect to the same two HS *x* CI controllers as the boot device. These two controllers must be connected through the same CI star coupler.
- The dump device directory structure must resemble the current system disk structure. The [SYS *n*.SYSEXE]SYSDUMP.DMP file will reside there, with the same boot time system root.

  Use AUTOGEN to create this file. In the MODPARAMS.DAT file, the following symbol prompts AUTOGEN to create the file:

  ```
  DUMPFILE_DEVICE = "$nnn$ddcuuuu"
  ```

  You can list only one device.
- The volume label can be up to 12 characters long. The ASCII string DOSD_DUMP must be part of this volume label. For example, valid volume labels are DOSD_DUMP, DOSD_DUMP_12, 12_DOSD_DUMP. The label is read and retained in a memory boot data structure.
- The dump device cannot be part of a volume set. VSI strongly recommends that the dump device also not be part of a shadow set.
- The dump device cannot be MSCP unit zero (0); only units 1 to 4095 (1 – FFF) are supported.

  You can designate the dump device as follows:

  - On VAX 7000 configurations, by using bits 16 through 27 of the DUMPSTYLE system parameter. Note that the DUMP_DEV environment variable that is provided on VAX 7000 configurations is not used by OpenVMS VAX.

- On configurations other than the VAX 7000, by using bits 16 through 27 of register 3 (R3). You can use this portion of the register to specify the dump device.

The *VSI OpenVMS System Management Utilities Reference Manual* and online help contain details about the DUMPSTYLE system parameter.

Note

To restore error log buffers when the system is rebooted after a system crash, the error logs must be saved on the system disk. For this purpose, AUTOGEN creates a SYSDUMP.DMP file on the system disk; the file is large enough to contain the maximum size of error log buffers.

2.8. Using SDA to Analyze the Contents of a Crash Dump

The System Dump Analyzer utility (SDA) lets you interpret the contents of the system dump file to investigate the probable causes of the crash. For information about analyzing a crash dump, refer to the *OpenVMS VAX System Dump Analyzer Utility Manual* or the *VSI OpenVMS Alpha System Analysis Tools Manual*.

If your system fails, use SDA to make a copy of the system dump file written at the time of the failure and contact your VSI support representative. For information about copying the system dump file, see *Section 2.11, "Copying System Dump Files to Tape or Disk"*.

2.8.1. Using SDA CLUE Commands to Analyze Crash Dump Files (Alpha Only)

SDA CLUE (Crash Log Utility Extractor) commands automate the analysis of crash dumps and maintain a history of all fatal bugchecks on a standalone system or cluster. You can use SDA CLUE commands in conjunction with SDA to collect and decode additional dump file information not readily accessible through standard SDA. You can also use SDA CLUE with Dump Off System Disk (DOSD) to analyze a system dump file that resides on a disk other than the system disk.

2.8.1.1. Understanding CLUE (Alpha Only)

On Alpha systems, SDA is automatically invoked by default when you reboot the system after a system failure. To better facilitate crash dump analysis, SDA CLUE commands automatically capture and archive summary dump file information in a CLUE listing file.

A startup command procedure initiates commands that:

- Invoke SDA
- Issue an SDA CLUE HISTORY command
- Create a listing file called CLUE$nodename_ddmmyy_hhmm.LIS

The CLUE HISTORY command adds a one-line summary entry to a history file and saves the following output from SDA CLUE commands in the listing file:

- Crash dump summary information
- System configuration
- Stack decoder
- Page and swap files
- Memory management statistics
- Process DCL recall buffer
- Active XQP processes
- XQP cache header

The contents of this CLUE list file can help you analyze a system failure.

If these files accumulate more space than the threshold allows (default 5000 blocks), the oldest files are deleted until the threshold limit is reached. This can also be customized using the CLUE$MAX_BLOCK logical name.

To inhibit the running of CLUE at system startup, define the logical CLUE$INHIBIT in the SYLOGICALS.COM file as /SYS TRUE.

It is important to remember that CLUE$nodename_ddmmyy_hhmm.LIS contains only an overview of the crash dump and does not always contain enough information to determine the cause of the crash. If you must do an in-depth analysis of the system crash, VSI recommends that you always use the SDA COPY command to save the dump file.

2.8.1.2. Displaying Data Using SDA CLUE Commands (Alpha Only)

Invoke CLUE commands at the SDA prompt as follows:

```
SDA> CLUE CONFIG
```

CLUE commands provide summary information of a crash dump captured from a dump file. When debugging a crash dump interactively, you can use SDA CLUE commands to collect and decode some additional information from a dump file, which is not easily accessible through standard SDA. For example, CLUE can quickly provide detailed XQP summaries.

You can also use CLUE commands interactively on a running system to help identify performance problems.

You can use all CLUE commands when analyzing crash dumps; the only CLUE commands that are not allowed when analyzing a running system are CLUE CRASH, CLUE ERRLOG, CLUE HISTORY, and CLUE STACK.

Refer to the OpenVMS Alpha System Analysis Tools Manual for more information about using SDA CLUE commands.

2.8.1.3. Using SDA CLUE with Dump Off System Disk (Alpha Only)

Dump off system disk (DOSD) allows you to write the system dump file to a device other than the system disk. For SDA CLUE to be able to correctly find the dump file to be analyzed after a system crash, perform the following steps:

1. Modify the command procedure SYS$MANAGER:SYCONFIG.COM to add the system logical name CLUE$DOSD_DEVICE to point to the device where the dump file resides. You need to supply only the physical or logical device name without a file specification.

2. Modify the command procedure SYS$MANAGER:SYCONFIG.COM to mount systemwide the device where the dump file resides. Otherwise, SDA CLUE cannot access and analyze the dump file.

In the following example, the dump file is placed on device 3DUA25, with the label DMP$DEV. You need to add the following commands to SYS$MANAGER:SYCONFIG.COM:

```
$ mount/system/noassist $3$dua25: dmp$dev dmp$dev
$ define/system clue$dosd_device dmp$dev
```

2.9. Using CLUE to Obtain Historical Information About Crash Dumps (VAX Only)

On VAX systems, the Crash Log Utility Extractor (CLUE) displays the contents of a **crash history file**. By examining the contents of the crash history file, you can understand and resolve the issues responsible for failures (crashes), and you might also obtain other useful data.

2.9.1. Understanding CLUE (VAX Only)

The crash history file, which is created and updated by CLUE, contains key parameters from crash dump files. Unlike crash dumps, which are overwritten with each system failure and are therefore typically available only for the most recent failure, the crash history file is a permanent record of system failures.

After a system fails and physical memory is copied to the crash dump file, CLUE automatically appends the relevant parameters to the file CLUE$OUTPUT:CLUE$HISTORY.DATA when the system is restarted. The remainder of this section describes how you can use CLUE to display the data it has collected; reference information about CLUE is available in the *VSI OpenVMS System Management Utilities Reference Manual*.

Note

The history file typically grows by about 10 to 15 blocks for each entry. You can limit the number of entries in the binary file by defining the logical name CLUE$MAX_ENTRIES to be the maximum number desired. When this number is reached, the oldest entries are deleted from the history file.

By default, operator shutdowns are recorded in the history file. You can exclude information from operator shutdowns in the history file by defining the logical name CLUE$EXCLUDE_OPERS as being TRUE, for example by including the following line in SYS$MANAGER:SYSTARTUP_VMS.COM:

```
$ DEFINE /SYSTEM CLUE$EXCLUDE_OPERS TRUE
```

2.9.2. Displaying Data Using CLUE (VAX Only)

To display data using CLUE, you must first define the following symbol:

```
$ CLUE :== $CLUE
```

After defining the symbol, you can use CLUE to display information by entering the following command:

```
$ CLUE/DISPLAY
CLUE_DISPLAY>
```

At the CLUE_DISPLAY> prompt, you can issue commands to perform the following actions:

- Use the DIRECTORY command to list failures that have occurred since a specified date, failures of a particular type, failures that contain a specified module, and failures that have a specified offset.

  For example, you can list all the failures in the history file using the DIRECTORY command, as follows:

  ```
  CLUE_DISPLAY> DIRECTORY
  ```

- Use the SHOW command to generate information similar to that obtained from certain commands in the System Dump Analyzer utility (SDA).

  For example, if you wanted complete information about the crash listed as crash number 7, the following SHOW command would provide the information:

  ```
  CLUE_DISPLAY> SHOW ALL 7
  ```

- Use the EXTRACT command to write the data from an entry to a file.

  For example, the following command writes the data from entry number 7 in the crash history file to a file named 15MAYCRASH.TXT:

  ```
  CLUE_DISPLAY> EXTRACT 7/OUTPUT=15MAYCRASH.TXT
  ```

For more information about CLUE commands, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

2.10. Saving the Contents of the System Dump File After a System Failure

If the system fails, it overwrites the contents of the system crash dump file and the previous contents are lost. For this reason, ensure that your system automatically analyzes and copies the contents of the system dump file each time the system reboots.

On Alpha systems, SDA is invoked by default during startup, and a CLUE list file is created. Generated by a set sequence of commands, the CLUE list file contains only an overview of the crash and might not

provide enough information to determine the cause of the crash. VSI, therefore, recommends that you always copy the system dump file.

Refer to the *VSI OpenVMS Alpha System Analysis Tools Manual* for information about modifying your site-specific command procedure to execute additional commands such as SDA COPY upon startup after a system failure.

On VAX systems, modify the site-specific startup command procedure SYSTARTUP_VMS.COM so that it invokes the System Dump Analyzer utility (SDA) when the system is booted.

Be aware of the following facts:

- When invoked from the site-specific startup procedure in the STARTUP process, SDA executes the specified commands only if the system is booting immediately after a system failure. If the system is rebooting after it was shut down with SHUTDOWN.COM or OPCCRASH.EXE, SDA exits without executing the commands.
- Although you can use the DCL command COPY to copy the dump file, the SDA command COPY is preferable because it copies only the blocks occupied by the dump and it marks the dump file as copied. The SDA COPY command is preferable also when the dump was written into the paging file, SYS$SYSTEM:PAGEFILE.SYS, because the SDA COPY command releases to the pager those pages occupied by the dump. For more information, see *Section 2.12, "Freeing Dump Information from the Page File"*.
- Because a system dump file can contain privileged information, protect copies of dump files from world read access. For more information about file protection, refer to the *VSI OpenVMS Guide to System Security*.
- System dump files have the NOBACKUP attribute, so the Backup utility (BACKUP) does not copy them unless you use the qualifier /IGNORE=NOBACKUP when invoking BACKUP. When you use the SDA command COPY to copy the system dump file to another file, the operating system does not automatically set the new file to NOBACKUP. If you want to set the NOBACKUP attribute on the copy, use the SET FILE command with the /NOBACKUP qualifier as described in the *VSI OpenVMS DCL Dictionary*.

Example

The SDA command COPY in the following example saves the contents of the file SYS$SYSTEM:PAGEFILE.SYS and performs some analysis of the file. Note that the COPY command is the final command because the blocks of the page file used by the dump are released as soon as the COPY command completes, and can be used for paging before any other SDA commands can be executed.

```
$ !
$ !      Print dump listing if system just failed
$ !
$ ANALYZE/CRASH_DUMP SYS$SYSTEM:PAGEFILE.SYS
    SET OUTPUT DISK1:SYSDUMP.LIS        ! Create listing file
    READ/EXECUTIVE                      ! Read in symbols for kernel
    SHOW CRASH                          ! Display crash information
    SHOW STACK                          ! Show current stack
    SHOW SUMMARY                        ! List all active processes
    SHOW PROCESS/PCB/PHD/REG            ! Display current process
    COPY SYS$SYSTEM:SAVEDUMP.DMP        ! Save system dump file
    EXIT
```

```
$ SET FILE/NOBACKUP SYS$SYSTEM:SAVEDUMP.DMP
```

2.11. Copying System Dump Files to Tape or Disk

If your system fails, make a copy of the contents of the system dump file and contact your VSI support representative. You can use the Backup utility (BACKUP) to create save sets containing system dump files on magnetic tape or disk. However, when using BACKUP to copy system dump files, you must specify the /IGNORE=(NOBACKUP,INTERLOCK) qualifier for the following reasons:

- By default, the system dump file has the NOBACKUP attribute, so it is not copied unless you specify /IGNORE=NOBACKUP.
- The system keeps an open channel to the system dump file, so the file is not copied unless you specify /IGNORE=INTERLOCK.

For more information about using BACKUP, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*. For information about BACKUP commands, see the *VSI OpenVMS System Management Utilities Reference Manual*.

VSI recommends that you use the following procedure to copy your system dump file:

1. Use the SDA command COPY to make a copy of the system dump file.
2. Use BACKUP to save that copy on tape or disk.

This procedure avoids issues with BACKUP qualifiers and reduces the amount of data written to tape or disk because the SDA command COPY copies only blocks in the system dump file that are actually used.

2.12. Freeing Dump Information from the Page File

If you use SYS$SYSTEM:PAGEFILE.SYS to store a system crash dump, you must later free the space occupied by the system dump for use by the pager. If you do not, your system might hang because of insufficient paging space.

Section 2.1.1, "Using the Page File to Store System Crash Dumps" explains when you might use the page file to store a system crash dump.

2.12.1. Freeing Dump Information on VAX and Alpha Systems

This section contains instructions for freeing dump information from the page file on VAX and Alpha systems.

How to Perform This Task on VAX Systems

On VAX systems, perform the following steps:

1. Invoke the System Dump Analyzer utility (SDA), specifying PAGEFILE.SYS as the target:

```
$ ANALYZE/CRASH_DUMP SYS$SYSTEM:PAGEFILE.SYS
```

2. Enter the SDA command COPY in the following format to copy the dump from SYS $SYSTEM:PAGEFILE.SYS to another file:

   ```
   COPY dump_filespec
   ```

   For example, to copy the system dump file off the system disk to a file called SAVEDUMP.DMP on DISK$USER5, enter the following command:

   ```
   SDA> COPY DISK$USER5:[DUMPS]SAVEDUMP.DMP
   ```

3. Enter the EXIT command to exit SDA.

4. Include the SDA commands entered in steps 1 and 2 in the site-specific startup command procedure SYSTARTUP_VMS.COM to free page space each time the system reboots.

Alternatively, to free the pages in the page file that are taken up by the dump without having to copy the dump elsewhere, enter the ANALYZE/CRASH_DUMP/RELEASE command. This command immediately releases the pages to be used for system paging, effectively deleting the dump. Note that this command does *not* allow you to analyze the dump before deleting it.

Example

The following commands, added to the SYSTARTUP_VMS.COM command procedure, copy the contents of the page file to a file named SAVEDUMP.DMP:

```
$ ANALYZE/CRASH_DUMP SYS$SYSTEM:PAGEFILE.SYS
  COPY DISK$USER5:[DUMPS]SAVEDUMP.DMP
  EXIT
$ SET FILE/NOBACKUP SYS$SYSTEM:SAVEDUMP.DMP
```

How to Perform This Task on Alpha Systems

On Alpha systems, as described in the *VSI OpenVMS Alpha System Analysis Tools Manual*, SDA is automatically invoked by default when the system is rebooted after a system failure.

To automatically save the system dump file, perform the following steps:

1. Create a SYS$MANAGER:SAVEDUMP.COM file; for example:

   ```
   !
   ! SDA command file, to be executed as part of the system
   ! bootstrap from within CLUE.  Commands in this file can
   ! be used to save the dump file after a system bugcheck, and
   ! to execute any additional SDA command.
   !
   READ/EXEC               ! Read in the executive images' symbol tables
   SHOW STACK              ! Display the stack
   COPY SAVEDUMP.DMP       ! Copy and save system dump file

   !
   ```

2. To point to your site-specific file, add a line such as the following one to the file SYS $MANAGER:SYLOGICALS.COM:

   ```
   $ DEFINE/SYSTEM CLUE$SITE_PROC SYS$MANAGER:SAVEDUMP.COM
   ```

In this example, the site-specific file is named SAVEDUMP.COM.

If the logical CLUE$INHIBIT has been defined, and SDA has not been automatically invoked during system startup, the pages in the page file that are taken up by the dump can be released using the ANALYZE/CRASH_DUMP/RELEASE command. This command immediately releases the pages to be used for system paging, effectively deleting the dump. Note that this command does *not* allow you to analyze the dump before deleting it.

For a discussion of logical names used by CLUE, refer to *VSI OpenVMS Alpha System Analysis Tools Manual*.

2.12.2. Usage Notes for Freeing Information on VAX and Alpha Systems

Because a system dump file can contain privileged information, protect copies of dump files from world read access.

To prevent the system from backing up the complete contents of the file, assign the NOBACKUP attribute to the file with the DCL command SET FILE/NOBACKUP.

Although you can also use the DCL command COPY to copy a dump file, VSI recommends that you use the SDA command COPY because SDA COPY performs the following actions:

- Copies only the blocks that the dump actually occupies.
- Releases for paging the pages that the dump occupies in the system's page file.

2.13. Installing Page and Swap Files

The system automatically installs the primary page and swap files located in SYS$SYSTEM. However, other page and swap files are not automatically installed. For this reason, if you create secondary page and swap files, you must also install them with the System Generation utility (SYSGEN). Note that SYSGEN INSTALL commands perform a different function than Install utility (INSTALL) commands.

2.13.1. Installing Interactively

1. Invoke SYSGEN by entering the following command:

   ```
   $ RUN SYS$SYSTEM:SYSGEN
   ```

2. Enter the SYSGEN command INSTALL as follows:

   For page files, use the following format:

   ```
   INSTALL file-spec/PAGEFILE
   ```

   For example:

   ```
   SYSGEN> INSTALL DUA2:[PAGE_SWAP]PAGEFILE_1.SYS/PAGEFILE
   ```

   For swap files, use the following format:

   ```
   INSTALL file-spec/SWAPFILE
   ```

For example:

```
SYSGEN> INSTALL DUA2:[PAGE_SWAP]SWAPFILE_1.SYS/SWAPFILE
```

3. To make sure the files are installed each time the system boots, edit SYS $MANAGER:SYPAGSWPFILES.COM to add the commands you entered in step 2. For more information, see *Section 2.13.2, "Installing in SYPAGSWPFILES.COM "*.

Example

The following example installs page and swap files interactively:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> INSTALL DUA2:[PAGE_SWAP]PAGEFILE_1.SYS/PAGEFILE
SYSGEN> INSTALL DUA2:[PAGE_SWAP]SWAPFILE_1.SYS/SWAPFILE
```

2.13.2. Installing in SYPAGSWPFILES.COM

Page and swap files other than SYS$SYSTEM:PAGEFILE.SYS and SYS$SYSTEM:SWAPFILE.SYS must be reinstalled each time the system boots. You can do this by adding the commands to install the files to the startup command procedure SYS$MANAGER:SYPAGSWPFILES.COM. The template file SYS$MANAGER:SYPAGSWPFILES.TEMPLATE includes comments that help explain how this file is used.

Before performing this task, you must have created the secondary files, as explained in *Section 2.15, "Creating and Modifying Page, Swap, and Dump Files"*.

For more information about SYPAGSWPFILES.COM, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

You can also use SATELLITE_PAGE.COM to install page and swap files on an OpenVMS Cluster satellite node's local disk. SATELLITE_PAGE.COM is created when you run CLUSTER_CONFIG.COM. For more information about installing page and swap files on a satellite node's local disk, refer to the *VSI OpenVMS Cluster Systems Manual* manual.

How to Perform This Task

1. Invoke any editor to edit SYS$MANAGER:SYPAGSWPFILES.COM.

2. If necessary, add a MOUNT command for each disk that holds a page or swap file. This is necessary because only the system disk is mounted at the time SYPAGSWPFILES.COM is invoked.

   For example:

   ```
   $ MOUNT/SYSTEM/NOASSIST DUA2: DISK_SYS2
   ```

   For information about the MOUNT command, refer to the *VSI OpenVMS DCL Dictionary*.

   The following commands, inserted before the MOUNT command, are also useful to determine if the disk is available before mounting. Note, however, that if the disk is broken and cannot mount, these commands will cause an infinite loop.

   ```
   $ LOOP1:
   $  ON WARNING THEN GOTO LOOP1
   ```

```
$  WAIT 0000 00:00:00.50
$  READY = F$GETDVI("device:","AVL")
$  IF READY .EQS. "FALSE" THEN GOTO LOOP1
```

where *device:* specifies the device name.

3. Add the following command to invoke SYSGEN:

```
$ RUN SYS$SYSTEM:SYSGEN
```

4. Add commands in the following format to SYPAGSWPFILES.COM to install the files each time the system boots.

   For page files, use the following format:

```
INSTALL file-spec/PAGEFILE
```

   For example:

```
INSTALL DUA2:[SYSTEM]PAGEFILE_1.SYS/PAGEFILE
```

   For swap files, use the following format:

```
INSTALL file-spec/SWAPFILE
```

   For example:

```
INSTALL DUA2:[SYSTEM]SWAPFILE_1.SYS/SWAPFILE
```

5. Add an EXIT command to exit SYSGEN:

```
EXIT
```

Example

The following example shows commands you might add to SYPAGSWPFILES.COM to install page and swap files named PAGEFILE_1.SYS and SWAPFILE_1.SYS located on the DUA2: device:

```
$ EDIT SYS$MANAGER:SYPAGSWPFILES.COM
  [add the following commands to SYPAGSWPFILES.COM:]
   .
   .
   .

$ MOUNT/SYSTEM/NOASSIST DUA2: DISK_SYS2
$ RUN SYS$SYSTEM:SYSGEN
  INSTALL DUA2:[SYSTEM]PAGEFILE_1.SYS /PAGEFILE
  INSTALL DUA2:[SYSTEM]SWAPFILE_1.SYS /SWAPFILE
  EXIT
```

2.14. Removing Page, Swap, and Dump Files

Caution

If you remove a system page, swap, or dump file, do not simply delete the file. The disk might become corrupted if you continue to use the system after you delete the files.

How to Perform This Task

1. Use the RENAME command to rename the file to be deleted.
2. Shut down and reboot the system.
3. Delete the file.
4. When you delete a file, make sure you remove from SYPAGSWPFILES.COM and MODPARAMS.DAT any command lines related to the file.

Example

```
$ RENAME DUA2:[SYSTEM]PAGEFILE_1.SYS; DUA2:[SYSTEM]JUNK.SYS;
$ @SYS$SYSTEM:SHUTDOWN.COM
.
[SHUTDOWN.COM shuts down and reboots the system]
[When the system reboots, log in]
.
$ DELETE DUA2:[SYSTEM]JUNK.SYS;
```

2.15. Creating and Modifying Page, Swap, and Dump Files

For performance or disk space reasons, you might want to create system page, swap, and dump files on disks other than the system disk. (Error log dump files, however, must remain on the system disk.)

The following sections explain how to perform this task:

Method	For More Information
Using AUTOGEN (recommended method)	*Section 2.15.1, "Using AUTOGEN (Recommended Method)"*
Using SWAPFILES.COM (for primary files only)	*Section 2.15.2, "Using SWAPFILES.COM"*
Using SYSGEN	*Section 2.15.3, "Using SYSGEN"*

2.15.1. Using AUTOGEN (Recommended Method)

You can direct AUTOGEN to create new system page, swap, and dump files by adding symbols to MODPARAMS.DAT to specify the name, location, and size of new files to be created and then running AUTOGEN. Before performing this task, you should understand AUTOGEN and its parameter file MODPARAMS.DAT. For more information about when to use AUTOGEN, see *Section 1.4, "Understanding the AUTOGEN Command Procedure"*. See *Section 1.4.4, "AUTOGEN Parameter File (MODPARAMS.DAT)"* for information about MODPARAMS.DAT.

AUTOGEN automatically calculates appropriate sizes for system page, swap, and dump files. It also modifies the files to the appropriate sizes and installs them. You can control sizes calculated by AUTOGEN by defining symbols in the file MODPARAMS.DAT. For more information, see *Section 2.15.1.2, "Controlling the Size of System Page, Swap, and Dump Files in MODPARAMS.DAT"*.

How to Perform This Task

To change the sizes of system page, swap, and dump files, execute AUTOGEN in two passes as follows:

1. Enter the following command to invoke a first pass of AUTOGEN. AUTOGEN displays its calculations for system file sizes to SYS$OUTPUT:

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

2. If the file sizes displayed in step 1 are inadequate, add symbols to MODPARAMS.DAT to control the size of files as explained in *Section 2.15.1.2, "Controlling the Size of System Page, Swap, and Dump Files in MODPARAMS.DAT"* and return to step 1.

3. When you are satisfied with the file sizes displayed in step 1, execute a second pass of AUTOGEN using the following command to install the modified system files when the system is rebooted:

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

4. Add commands to the site-specific startup command procedure SYPAGSWPFILES.COM to make sure the files are installed each time the system boots. For instructions, see *Section 2.13, "Installing Page and Swap Files"*.

2.15.1.1. Controlling the Location of System Page, Swap, and Dump Files

Add the following symbols to MODPARAMS.DAT to specify the names and locations of the page and swap files to be created:

Definition	For Page Files	For Swap Files	For Dump Files
File name and location	PAGEFILE *n*_NAME = *"file-spec"*	SWAPFILE *n*_NAME = *"file-spec"*	DUMPFILE_DEVICE = *"device"*

where:

- *n* specifies the page or swap file. Refer to the primary page and swap files by specifying a value of 1 for *n*; refer to subsequent files by specifying increasingly higher integer values for *n*. For example, to refer to a secondary page or swap file, specify a value of 2 for *n*.
- *file-spec* specifies the full file specification of the file to be created, and should be within quotation marks ("").
- *device* specifies the disk name to be used, and should be within quotation marks ("").

2.15.1.2. Controlling the Size of System Page, Swap, and Dump Files in MODPARAMS.DAT

You can add information to the AUTOGEN parameter file MODPARAMS.DAT to control the sizes that AUTOGEN calculates for system page, swap, and dump files. If you do not supply system file size

information in MODPARAMS.DAT, AUTOGEN performs default size calculations for page, swap, and dump files.

You can define symbols in MODPARAMS.DAT to specify either of the following items:

Size to Be Specified	For More Information
Total desired size for all page or swap files on a system (not valid for the system dump files)	*Table 2.4, "Symbols for Controlling the Total Size of Page, Swap, System Dump, or Error Log Dump File Space"*
Sizes for *individual* page, swap, or dump files	*Table 2.5, "Symbols for Controlling the Size of Individual Page and Swap Files"*

Note

You cannot specify sizes for both total and individual files. AUTOGEN issues a warning if conflicting symbol definitions exist in MODPARAMS.DAT.

For page and swap files, AUTOGEN generally manipulates the primary files SYS $SYSTEM:PAGEFILE.SYS and SYS$SYSTEM:SWAPFILE.SYS *only* if you have no other page and swap files. If you have secondary files, AUTOGEN manipulates the secondary files and excludes primary files. However, in some instances, AUTOGEN might modify the size of the primary page and swap files.

On VAX systems, for system dump files, AUTOGEN manipulates the size of only one file: the system dump file on the system disk if no DUMPFILE_DEVICE is given, or the system dump file on the specified device if DUMPFILE_DEVICE is specified.

On VAX systems, AUTOGEN always creates a minimal SYSDUMP.DMP file on the system disk for error log buffers if DUMPFILE_DEVICE is specified.

On Alpha systems, AUTOGEN only manipulates the size of the error log dump file on the system disk.

If you do not want AUTOGEN to change the sizes of the primary files, specify the following symbols in MODPARAMS.DAT:

```
PAGEFILE =   0
SWAPFILE =   0
DUMPFILE =   0
ERRLOGDUMP = 0 ! Alpha only
```

These symbols direct AUTOGEN to ignore the primary page, swap, and dump files when calculating sizes.

If the creation or extension of a system page, swap, or dump file would cause the target disk to become more than 95 percent full, AUTOGEN issues a warning and does not perform the operation.

On Alpha systems, however, the 95 percent rule does not apply to the error log dump file, SYS $ERRLOG.DMP. This file is created if the disk can hold it.

You can use AUTOGEN to create a page, swap, or dump file that is smaller than the current version of the file. After you have booted and begun using the new file, remember to use the DCL command PURGE to reclaim the disk space from the old version of the file.

To determine the current sizes of installed page and swap files, enter the DCL command SHOW MEMORY/FILES. If you increased the size of any of these files and have not rebooted, this command displays the original sizes. Use the DIRECTORY command to determine the size of dump files.

Note

AUTOGEN does not change file sizes if you specify a value of 0 or a value that is within 10 percent of the current size.

Table 2.4, "Symbols for Controlling the Total Size of Page, Swap, System Dump, or Error Log Dump File Space" lists the symbols you can define in MODPARAMS.DAT to control the *total* size of page file, swap file, system dump file, or error log dump file space.

Table 2.4. Symbols for Controlling the Total Size of Page, Swap, System Dump, or Error Log Dump File Space

Operation	Page File Symbol	Swap File Symbol	Dump File Symbol	Error Log File Symbol
To define the total amount of space	PAGEFILE = *n*[1]	SWAPFILE = *n* [1]	DUMPFILE = *n* [1]	ERRLOGDUMP = *n* [1]
To increase total size	ADD_PAGEFILE = *n*	ADD_SWAPFILE = *n*	ADD_DUMPFILE = *n*	ADD_ERRLOGDUMP = *n*
To specify maximum total size	MAX_PAGEFILE = *n*	MAX_SWAPFILE = *n*	MAX_DUMPFILE = *n*	MAX_ERRLOGDUMP = *n*
To specify minimum total size	MIN_PAGEFILE = *n*	MIN_SWAPFILE = *n*	MIN_DUMPFILE = *n*	MIN_ERRLOGDUMP = *n*

[1] *n* is the total size, in blocks. If *n* is 0, the corresponding AUTOGEN section is skipped. For page and swap files, if *n* is not 0 and no secondary files exist, AUTOGEN applies the value to primary files. If *n* is not 0, and secondary files exist, AUTOGEN applies any change evenly across all secondary page or swap files but, in most cases, does not change primary files. For dump files, if *n* is not 0, AUTOGEN applies the value to the dump file on the system disk if no DUMPFILE_DEVICE is given or the dump file on the specified device if a DUMPFILE_DEVICE is given.

Table 2.5, "Symbols for Controlling the Size of Individual Page and Swap Files" lists the symbols you can define in MODPARAMS.DAT to control the size of *individual* files.

Table 2.5. Symbols for Controlling the Size of Individual Page and Swap Files

Operation	Page File Symbol [1]	Swap File Symbol [1]
To specify file size	PAGEFILE *n*_SIZE = *block-size*	SWAPFILE *n*_SIZE = *block-size*
To increase file size	ADD_PAGEFILE *n*_SIZE = *block-size*	ADD_SWAPFILE *n*_SIZE = *block-size*

Operation	Page File Symbol [1]	Swap File Symbol [1]
To specify maximum file size	MAX_PAGEFILE *n*_SIZE = *block-size*	MAX_SWAPFILE *n*_SIZE = *block-size*
To specify minimum file size	MIN_PAGEFILE *n*_SIZE = *block-size*	MIN_SWAPFILE *n*_SIZE = *block-size*

[1]For *n*, specify an integer that indicates the page or swap file. Refer to the primary page and swap files by specifying a value of 1 for *n*; refer to subsequent files by specifying increasingly higher integer values for *n*. For example, to refer to a secondary page or swap file, specify a value of 2 for *n*. For *block-size*, specify the size in blocks.

Examples

1. The following line in MODPARAMS.DAT specifies that all page file space should total 100,000 blocks:

   ```
   PAGEFILE = 100000
   ```

   If you had only a primary page file, the resulting size of that file would be 100,000 blocks. If you had multiple page files, the difference between the total current size and the total new size would be spread across secondary files. For example, if you specified PAGEFILE = 100000, the changed page file sizes would be as follows:

File	Original Size (in Blocks)	Resulting Size (in Blocks)
Primary page file	10,000	10,000
Secondary page file 1	30,000	45,000
Secondary page file 2	30,000	45,000

2. To direct AUTOGEN to set the primary page file size to 10,000 blocks, use the symbol definition:

   ```
   PAGEFILE1_SIZE = 10000
   ```

3. To direct AUTOGEN to create a new secondary swap file named PAGED$:[PAGESWAP]SWAPFILE.SYS that holds 30,000 blocks, use the symbol definitions:

   ```
   SWAPFILE2_NAME = "PAGED$:[PAGESWAP]SWAPFILE.SYS"
   MIN_SWAPFILE2_SIZE = 30000
   ```

2.15.2. Using SWAPFILES.COM

VSI recommends that you use AUTOGEN to change sizes of system page, swap, and dump files. However, you can use the command procedure SYS$UPDATE:SWAPFILES.COM to change the size of *primary* system page, swap, and dump files. SWAPFILES.COM shows you the current size of the system page, swap, and dump files before you change the sizes.

If you change the sizes of system page, swap, or dump files, you must edit MODPARAMS.DAT to specify the new sizes, as explained in *Section 2.15.1.2, "Controlling the Size of System Page, Swap, and Dump Files in MODPARAMS.DAT"*. If you do not specify the new sizes in MODPARAMS.DAT, AUTOGEN resizes the files next time it runs.

The procedure displays the sizes of the current system page, swap, and dump files in SYS$SYSTEM, and the amount of space remaining on the system disk. It then allows you to enter new sizes, or keep the existing sizes for these files. If you specify a size that is larger than that of an existing file, the procedure automatically extends the size of a page or dump file. If you specify a smaller size for a system page, swap, or dump file, a new version of the file is created.

How to Perform This Task

1. Enter the following command to invoke the command procedure:

   ```
   $ @SYS$UPDATE:SWAPFILES.COM
   ```

   The system displays the current files found in SYS$SYSTEM and their sizes. For example:

   ```
   Current file sizes are:

   Directory SYS$SYSROOT:[SYSEXE]

   PAGEFILE.SYS;1    16384
   SYSDUMP.DMP;1      4128
   SWAPFILE.SYS;1     3072

   Total of 3 files, 23584 blocks.

   There are 128741 available blocks on SYS$SYSDEVICE.
   ```

2. In response to the following prompt, type the desired size, in blocks, for the page file. To keep the same size, press Return:

   ```
   Enter new size for page file:
   ```

3. In response to the following prompt, type the desired size, in blocks, for the dump file. To keep the same size, press Return:

   ```
   Enter new size for system dump file:
   ```

4. In response to the following prompt, type the desired size, in blocks, for the swap file. To keep the same size, press Return:

   ```
   Enter new size for swap file:
   ```

5. Shut down and reboot the system to use the new files.

6. After the system reboots, purge obsolete copies of the files. Do not delete the old files until the system reboots.

7. Edit MODPARAMS.DAT to include the new file sizes, as explained in *Section 2.15.1.2, "Controlling the Size of System Page, Swap, and Dump Files in MODPARAMS.DAT"*. If you do not specify the new sizes in MODPARAMS.DAT, AUTOGEN will automatically resize the files the next time it runs.

Example

```
$ @SYS$UPDATE:SWAPFILES.COM
To leave a file size at its current value type a
carriage return in response to its size prompt.
Current file sizes are:

Directory SYS$SYSROOT:[SYSEXE]

PAGEFILE.SYS;1    100000
SYSDUMP.DMP;1      28000
SWAPFILE.SYS;1     33000
```

```
Total of 3 files, 161000 blocks.

There are 128741 available blocks on SYS$SYSDEVICE.

Enter new size for page file: [Return]
Enter new size for system dump file: 30000
%SYSGEN-I-EXTENDED, SYS$SYSROOT:[SYSEXE]SYSDUMP.DMP;1 extended
Enter new size for swap file: [Return]


*****************************************************************
*  Please reboot in order for the new files to be used by the system. *
*  After rebooting, purge obsolete copies of the files.              *
*  DO NOT delete the old files until after the reboot.               *
*****************************************************************
```

2.15.3. Using SYSGEN

VSI recommends that you use AUTOGEN to create and change page, swap, and dump files. AUTOGEN invokes the System Generation utility (SYSGEN) to create or change the files. However, in an emergency, you can use SYSGEN to directly change the size of page, swap and dump files. For example, if you see that page file space is becoming dangerously low, you might use SYSGEN to quickly add page file space to prevent the system from hanging.

Note

VAX: System parameters SWPFILCNT and PAGFILCNT limit the number of swap and page files that the system installs. Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for more information.

Alpha: OpenVMS Alpha supports a maximum of 254 page and or swap files on a system. System parameters SWPFILCNT and PAGFILCNT are not used on Alpha systems.

How to Perform This Task

1. Determine the location and appropriate size of the files. For information, see *Section 2.4, "Manually Calculating Appropriate Sizes for Dump, Page, and Swap Files"*.
2. Invoke SYSGEN and enter the CREATE command in the following format:

   ```
   CREATE file-spec/SIZE=block-count
   ```

   where:

   file-spec specifies the full file specification.

   block-count specifies the size of the file in blocks.

   If the file you specify already exists and the size you specify is larger than the existing file, the command extends the existing file. If the file you specify already exists and the size you specify is smaller than the existing file, the command creates a new file of the specified size.

   For example, the following command extends the existing, smaller primary page file PAGEFILE.SYS:

```
SYSGEN> CREATE PAGEFILE.SYS/SIZE=100000
```

For more information about the SYSGEN command CREATE, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

Note

Frequent file creation and deletion can cause the free space on a disk to become severely fragmented. SYSGEN issues a HEADERFULL warning message if it determines that the creation or extension of a system file would cause that file to become fragmented enough to render the system unbootable. If this occurs, VSI recommends that you back up and restore your system disk to consolidate the free space on the volume into one contiguous area. For more information, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

After you restore the disk, retry the SYSGEN operation. When SYSGEN issues a warning message, the file might be somewhat larger, but not as large as the value specified in the CREATE command.

---

3. Use the following table to determine if you should reboot to use the new or modified file:

Type	Change	Reboot Required?
Primary page, swap, system dump, or error log dump file[1]	New file	Yes
	Extended file	Yes
Secondary page or swap file	New file	No[2]
	Extended file	Yes
Alternate (DOSD) dump file (Alpha)	New file	No
	Extended file	No
Alternate (DOSD) dump file (VAX)	New file	Yes
	Extended file	Yes

[1]Primary page, swap, and dump files are SYS$SPECIFIC:[SYSEXE] PAGEFILE.SYS, SWAPFILE.SYS, SYSDUMP.DMP; and SYS$ERRLOG.DMP.

[2]Although rebooting the system is unnecessary, you must install secondary files before the system can use them. For more information, see *Section 2.13, "Installing Page and Swap Files"*.

4. If you create a new version of the file, purge the old version *after* the system reboots.

5. Add commands to the site-specific startup command procedure SYPAGSWPFILES.COM to make sure the files are installed each time the system boots. For instructions, see *Section 2.13, "Installing Page and Swap Files"*.

6. If you do not want AUTOGEN to resize the files according to its calculations, edit MODPARAMS.DAT to specify the sizes of these files. Follow the instructions in *Section 2.15.1.2, "Controlling the Size of System Page, Swap, and Dump Files in MODPARAMS.DAT"*.

Example

The commands in the following example extend the existing files PAGEFILE.SYS, SWAPFILE.SYS, and SYSDUMP.DMP to the specified sizes:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> CREATE PAGEFILE.SYS/SIZE=100000
```

```
%SYSGEN-I-EXTENDED, SYS$SYSROOT:[SYSEXE]PAGEFILE.SYS;1 extended
SYSGEN> CREATE SWAPFILE.SYS/SIZE=30000
%SYSGEN-I-EXTENDED, SYS$SYSROOT:[SYSEXE]SWAPFILE.SYS;1 extended
SYSGEN> CREATE SYSDUMP.DMP/SIZE=33000
%SYSGEN-I-EXTENDED, SYS$SYSROOT:[SYSEXE]SYSDUMP.DMP;1 extended
SYSGEN> EXIT
```

2.16. Understanding Process Dumps

When a single process fails but the operating system is still running, the system can create a **process dump** that contains information about the process to assist in determining what caused the process to fail.

By default, process dumps are written to the current default directory of the user. You can override this by defining the logical name SYS$PROCDMP to identify an alternate directory path. Note that the name of the process dump file is always the same as the name of the main image active at the time the process dump is written, with the file extension .DMP.

On Alpha systems, a process dump is either complete or partial. A **complete process dump** contains all of process space and all process-pertinent data from system space. A **partial process dump** contains only user-readable data from process space and only those data structures from system space that are not deemed sensitive. Privileged or protected data, such as an encryption key in third-party software, might be considered sensitive.

On Alpha systems, you can force a dump to be written for another process with the DCL command SET PROCESS/DUMP=NOW *process-spec*. This command causes the contents of the address space occupied by process-spec to be written immediately to the file named *image-name*.DMP in the current directory of *process-spec*.

For more information about the DCL command SET PROCESS/DUMP, refer to the *VSI OpenVMS DCL Dictionary*.

2.16.1. Understanding Privileged Users and Access to Process Dumps (Alpha Only)

For this discussion, a privileged user is one who satisfies one of the following conditions:

- Has one or more of the privileges CMKRNL, CMEXEC, SYSPRV, READALL, or BYPASS
- Is a member of a system UIC group (by default [10,n] or lower). Such users are treated as though they hold SYSPRV privilege.

Holders of CMKRNL or CMEXEC can write complete process dumps. Holders of any of the other privileges mentioned above can read a process dump wherever it has been written.

In general, nonprivileged users should not be able to read complete process dumps, and by default they cannot do so. However, certain situations require that a nonprivileged user be able to read a complete process dump. Other situations require that a nonprivileged user be able to create a complete process dump but be able to read only a partial process dump.

Rights identifier IMGDMP$READALL enables a nonprivileged user to read a complete process dump. Rights identifier IMGDMP$PROTECT protects a complete process dump from being read by the nonprivileged user that created the process dump. These rights identifiers are created during the

installation of OpenVMS by the image SYS$SYSTEM:IMGDMP_RIGHTS.EXE, which is also run automatically during system startup to ensure that these rights identifiers exist with the correct values and attributes.

If these rights identifiers have been deleted, you can run SYS$SYSTEM:IMGDMP_RIGHTS.EXE to recreate them. For example:

```
$ RUN SYS$SYSTEM:IMGDMP_RIGHTS
%PROCDUMP-I-CREATED, rights identifier IMGDMP$READALL successfully created
%PROCDUMP-I-CREATED, rights identifier IMGDMP$PROTECT successfully created
```

Note that IMGDMP$READALL has no attributes, but IMGDMP$PROTECT is created with the RESOURCE attribute.

2.16.2. Granting Access to Process Dumps (Alpha Only)

To allow a nonprivileged user to write and read complete process dumps, grant the rights identifier IMGDMP$READALL to the user. If the IMGDMP$READALL rights identifier does not exist, run the image SYS$SYSTEM:IMGDMP_RIGHTS.EXE to create it (see *Section 2.16.1, "Understanding Privileged Users and Access to Process Dumps (Alpha Only)"*). Then use AUTHORIZE to grant the rights identifier to the user. For example:

```
$ DEFINE /USER SYSUAF SYS$SYSTEM:SYSUAF.DAT  !if necessary
$ RUN SYS$SYSTEM:AUTHORIZE
UAF> GRANT /IDENTIFIER IMGDMP$READALL <user>
UAF> EXIT
```

Note that the user must log out and log in again to be able to exercise the rights identifier. A nonprivileged user with rights identifier IMGDMP$READALL can read and write complete process dumps without restriction.

2.16.3. Restricting Access to Process Dumps (Alpha Only)

You can allow a nonprivileged user to write a complete process dump and at the same time prevent the user from reading that process dump. To do so, perform the following steps:

1. If the IMGDMP$PROTECT rights identifier does not exist, run the image SYS$SYSTEM:IMGDMP_RIGHTS.EXE to create it (see *Section 2.16.1, "Understanding Privileged Users and Access to Process Dumps (Alpha Only)"*).

2. Create a protected directory with rights identifier IMGDMP$PROTECT. For example:

   ```
   $ CREATE /DIRECTORY DKA300:[PROCDUMPS] -
       /PROTECTION=(S:RWE,O:RWE,G,W) /OWNER_UIC=IMGDMP$PROTECT
   $ SET SECURITY DKA300:[000000]PROCDUMPS.DIR -
       /ACL=((DEFAULT_PROTECTION,SYSTEM:RWED,OWNER:RWED,GROUP:,WORLD:), -
         (IDENTIFIER=IMGDMP$PROTECT,ACCESS=READ+WRITE), -
         (IDENTIFIER=IMGDMP$PROTECT,OPTIONS=DEFAULT, -
          ACCESS=READ+WRITE+EXECUTE+DELETE+CONTROL), -
         (CREATOR,ACCESS=NONE))
   ```

3. Define the executive-mode logical name SYS$PROTECTED_PROCDMP to point to the protected directory. For example:

```
$ DEFINE /SYSTEM /EXECUTIVE_MODE SYS$PROTECTED_PROCDMP DKA300:
[PROCDUMPS]
```

4. If DISKQUOTA is to be used on the disk containing the protected directory, specify the maximum disk space to be used for process dumps. For example:

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> DISKQUOTA CREATE /DEVICE=DKA300  ! if necessary
SYSMAN> DISKQUOTA ENABLE /DEVICE=DKA300  ! if necessary
SYSMAN> DISKQUOTA ADD IMGDMP$PROTECT /DEVICE=DKA300 /PERMQUOTA=10000
SYSMAN> DISKQUOTA REBUILD /DEVICE=DKA300  ! if necessary
SYSMAN> EXIT
```

Warning

Do not grant IMGDMP$PROTECT to any user. It is granted and revoked as needed by SYS $SHARE:IMGDMP.EXE from executive mode while writing a process dump. If you grant it permanently to a user, then that user has access to all process dumps written to the protected directory.

You can choose to set up additional ACLs on the protected directory to further control which users are allowed to read and write process dumps there.

Note that to take a process dump when the image is installed with elevated privileges or belongs to a protected subsystem, the user must hold CMKRNL privilege, and is by definition a privileged user (see *Section 2.16.1, "Understanding Privileged Users and Access to Process Dumps (Alpha Only)"*).

Chapter 3. Performance Considerations

This chapter introduces the basic concepts of performance management. For more detailed information, refer to *OpenVMS Performance Management Manual*.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Knowing your work load	*Section 3.2, "Knowing Your Work Load"*
Choosing a work load management strategy	*Section 3.3, "Choosing a Work Load Management Strategy"*
Distributing the work load	*Section 3.4, "Distributing the Work Load"*
Predicting when tuning is required	*Section 3.6, "Predicting When Tuning Is Required"*
Evaluating tuning success	*Section 3.7, "Evaluating Tuning Success"*
Choosing performance options	*Section 3.8, "Choosing Performance Options"*
Installing images with the Install utility (INSTALL)	*Section 3.10, "Using INSTALL to Install Known Images"*
Reserving memory for specific uses (Alpha only)	*Section 3.11, "Reserved Memory Registry"*

This chapter explains the following concepts:

Concept	Section
Performance management	*Section 3.1, "Understanding Performance Management"*

Concept	Section
System tuning	*Section 3.5, "Understanding System Tuning"*
Images and known images	*Section 3.10.1, "Understanding Images and Known Images"*
Known file lists	*Section 3.10.2, "Understanding Known File Entries"*
Attributes of known images	*Section 3.10.3, "Understanding Attributes You Can Assign to Known Images"*

3.1. Understanding Performance Management

Performance management means optimizing your hardware and software resources for the current work load. This task entails several distinct but related activities:

- Acquiring a thorough familiarity with your work load and an understanding of how that work load exercises the system's resources. This knowledge, combined with an appreciation of the operating system's resource management mechanisms, will enable you to establish realistic standards for system performance in areas such as the following ones:
  - Interactive and batch throughput
  - Interactive response time
  - Batch job turnaround time
- Routinely monitoring system behavior to determine if, when, and why a given resource is approaching capacity.
- Investigating reports of degraded performance from users.
- Planning for changes in the system work load or hardware configuration and being prepared to make any necessary adjustments to system values.
- Performing, after installation, certain optional system management operations.

3.2. Knowing Your Work Load

One of the most important assets that a system manager brings to any performance evaluation is an understanding of the normal work load and behavior of the system. Each system manager must assume the responsibility for understanding the system's work load sufficiently to be able to recognize normal and abnormal behavior; to predict the effects of changes in applications, operations, or usage; and to

recognize typical throughput rates. The system manager should be able to answer such questions as the following ones:

- What is the typical number of users on the system at any given time of day?
- What is the typical response time for various tasks for this number of users, at any given hour of operation?
- What are the peak hours of operation?
- Which jobs typically run at which time of day?
- Which commonly run jobs are intensive consumers of the CPU, memory, and disk space?
- Which applications involve the most image activations?
- Which parts of the system software, if any, have been modified or user-written, such as device drivers?
- Do any known, or anticipated, system bottlenecks exist?

If you are new to the OpenVMS operating system or to system management, you should observe system operation using the following tools:

- Monitor utility
- Accounting utility
- SHOW commands (available through DCL)

The *OpenVMS Performance Management Manual* provides detailed procedures for using the Monitor utility and, to a lesser extent, other operating system tools to observe and evaluate system performance. Also, the *VSI OpenVMS System Management Utilities Reference Manual* provides reference information about using the Monitor utility.

Over time you will learn about metrics such as the typical page fault rate for your system, the typical CPU usage, the normal memory usage, and typical modes of operation. You will begin to see how certain activities affect system performance and how the number of users or the time of day affects some of the values.

As you continue to monitor your system, you will come to know what range of values is acceptable, and you will be better prepared to use these same tools, together with your knowledge, to detect unusual conditions. Routine evaluation of the system is critical for effective performance management. The best way to avoid problems is to anticipate them; you should not wait for problems to develop before you learn how the system performs.

You can learn more about your system's operation if you use the Monitor and Accounting utilities on a regular basis to capture and analyze certain key data items. By observing and collecting this data, you will also be able to see usage trends and predict when your system may reach its capacity.

You should also understand that system resources are used by system management tools. Be careful, therefore, in selecting the items you want to measure and the frequency with which you collect the data. If you use the tools excessively, the consumption of system resources to collect, store, and analyze the data can distort your picture of the system's work load and capacity. The best approach is to have a plan for collecting and analyzing the data.

3.3. Choosing a Work Load Management Strategy

System performance is directly proportional to the efficiency of work load management. Each installation must develop its own strategy for work load management. Before adjusting any system values, make sure you resolve the following issues:

- Does the work load “peak” at a particular time of day, that is, is it noticeably heavier than at other times?
- Can the work load be better balanced? Perhaps some voluntary measures can be adopted by users, after appropriate discussion.
- Could some jobs be run better as batch jobs, preferably during nonpeak hours?
- Have primary and secondary hours of operation been employed with users? If not, could system performance benefit by adopting this practice? If the primary and secondary hours are in use, are the choices of hours the most appropriate for all users? (Plan to review this issue every time you either add or remove users or applications, to ensure that the desired balance is maintained.)
- Can future applications be designed to work around any known or expected system bottlenecks? Can present applications be redesigned somewhat, for the same purpose? (Refer to the *Guide to OpenVMS File Applications* [https://docs.vmssoftware.com/guide-to-openvms-file-applications/].)
- Are you making the best use of the code-sharing ability that the operating system offers? Code sharing provides an excellent means to conserve memory and improve performance over the life of the system.

3.4. Distributing the Work Load

You should distribute the work load as evenly as possible over the time your system is running. Although the work schedule for your site may make it difficult to schedule interactive users at optimum times, the following techniques may be helpful:

- Run large jobs as batch jobs—Establish a site policy that encourages the submission of large jobs on a batch basis. Regulate the number of batch streams so that batch usage is high when interactive usage is low. You might also want to use DCL command qualifiers to run batch jobs at lower priority, adjust the working set sizes, or control the number of concurrent jobs. For information about setting up your batch environment, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.
- Restrict system use—Do not permit more users to log in at one time than the system can support with an adequate response time. You can restrict the number of interactive users with the DCL command SET LOGINS/INTERACTIVE. You can also control the number of concurrent processes with the MAXPROCESSCNT system parameter, and the number of remote terminals allowed to access the system at one time with the RJOBLIM system parameter. See *Section 1.5, "Modifying System Parameters with AUTOGEN"* for information about modifying system parameters. Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for descriptions of all system parameters.

  You might also restrict use of the system by groups of users to certain days and hours of the day. You can use the Authorize utility to define the permitted login hours for each user. In particular, refer

to the AUTHORIZE qualifier /PRIMEDAYS. For more information, see the *VSI OpenVMS System Management Utilities Reference Manual*.

You can use the DCL command SET DAY to override the conventional day of the week associations for primary and secondary days. For example, you might want to specify a primary day of the week as a secondary day when it is a holiday.

- Design applications to reduce demand on binding resources—If you know where your system bottlenecks are or where they will likely occur in the near future, you can distribute the work load more evenly by planning usage that minimizes demand on any bottleneck points. (Refer to the *Guide to OpenVMS File Applications* [https://docs.vmssoftware.com/guide-to-openvms-file-applications/].)

3.5. Understanding System Tuning

Tuning is the process of altering various system values to improve *overall* performance possible from any given configuration and work load. However, the process does not include the acquisition and installation of additional memory or devices, although in many cases such additions (when made at the appropriate time) can vastly improve system operation and performance.

On most systems, the work load is constantly changing. System parameters that produce optimal performance at one time may not produce optimal performance a short time later as the work load changes. Your goal is to establish values that produce acceptable performance under all of the changing work load conditions.

Before you undertake any action, you must recognize that the following sources of performance problems cannot be cured by adjusting system values:

- Improper operation
- Unreasonable performance expectations
- Insufficient memory for the applications attempted
- Inadequate hardware configuration for the work load, such as too slow a processor, too few buses for the devices, too few disks, and so forth
- Improper device choices for the work load, such as using disks with insufficient speed or capacity
- Hardware malfunctions
- Poor application design
- Allowing one process to consume all available resources

When you make adjustments, you normally select a very small number of values for change, based on a careful analysis of the behavior you observed. You control system resources by tuning the values of two types of parameters:

Parameter Type	Description
System parameters	The values set for system parameters control system resources on a systemwide basis. The AUTOGEN command procedure automatically sets system parameters to appropriate values for your system configuration. AUTOGEN can also record

Parameter Type	Description
	feedback from a running system to adjust those parameters based on the system's work load. The *OpenVMS Performance Management Manual* describes how to select the parameters and new values that are likely to produce the desired changes. *Section 1.5, "Modifying System Parameters with AUTOGEN"* explains how to use AUTOGEN to modify system parameter values.
UAF limits and quotas	The values set for limits and quotas in each user authorization file (UAF) record control system resources on a per-user basis. To control these values, use the Authorize utility. For information, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

Before you undertake any tuning operation, be sure you are familiar with the resource management mechanisms described in the *OpenVMS Performance Management Manual*. Understand the nature of system values before adjusting them. Without the proper level of understanding, you might degrade, rather than improve, overall performance.

3.6. Predicting When Tuning Is Required

Under most conditions, tuning is rarely required for OpenVMS systems. The AUTOGEN command procedure, which is included in the operating system, establishes initial values for all the configuration-dependent system parameters so that they match your particular configuration. For information about AUTOGEN, see *Section 1.4, "Understanding the AUTOGEN Command Procedure"*.

Additionally, the system includes features that, in a limited way, permit it to adjust itself dynamically during operation. That is, the system detects the need for adjustment in certain areas, such as the nonpaged dynamic pool, the working set size, and the number of pages on the free and modified page lists. The system makes rough adjustments in these areas automatically. As a result, these areas can grow dynamically, as appropriate, during normal operation.

Experience has shown that the most common cause of disappointment in system performance is insufficient hardware capacity. Once the demand on a system exceeds its capacity, adjusting system values will not result in any significant improvements, simply because such adjustments are a means of trading off or juggling existing resources.

Although tuning is rarely required, you should recognize that system tuning may be needed under the following conditions:

- If you have adjusted your system for optimal performance with current resources and then acquire new capacity, you must plan to compensate for the new configuration. In this situation, the first and most important action is to execute the AUTOGEN command procedure.
- If you anticipate a dramatic change in your work load, you should expect to compensate for the new work load.

3.7. Evaluating Tuning Success

Whenever you adjust your system, you should monitor its behavior afterward to be sure that you have obtained the desired results. To observe results, use the Monitor utility and the various forms of the DCL command SHOW. Refer to the *VSI OpenVMS DCL Dictionary* for detailed information about the SHOW

command. See *Section 6.9.2, "Invoking MONITOR"* for information about using MONITOR. Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for detailed descriptions of MONITOR commands.

For example, you might consider running some programs (whose results you believe are fixed and reproducible) at the same time that you run your normal work load. If you run the programs and measure their running times under nearly identical work load conditions both before and after your adjustments, you can obtain a basis for comparison.

However, when applying this technique, remember to take the measurements under very similar work load conditions. Also, remember that this test alone does not provide conclusive proof of success. The possibility always exists that your adjustments may have favored the performance of the image you are measuring—to the detriment of other images. Therefore, in all cases, continue to observe system behavior closely for a time after you make any changes.

3.8. Choosing Performance Options

The following list describes optional system management operations normally performed after installation. These operations often result in improved overall performance. Choose the options that are appropriate for your site. Not all options are appropriate at every site.

- Expand system libraries — Most large libraries ship on the operating system in a data-reduced (compressed) format to conserve disk space. Unless the files are expanded (decompressed), the system must dynamically expand them each time they are accessed. The resulting performance slowdown is especially noticeable during link operations and requests to online help. If your system has sufficient disk space, expanding the libraries improves both CPU and elapsed time performance. For details about expanding system libraries and using the LIBDECOMP.COM command procedure, see *Section 3.9, "Expanding System Libraries"*.

- Disable file system high-water marking—This security feature is set by default when a volume is initialized to guarantee that users cannot read data they have not written.

  For nonshared sequential files, the performance impact of high-water marking is minimal. However, for files of nonsequential format, high-water marking creates some overhead; the system erases the previous contents of the disk blocks allocated every time a file is created or extended.

  Disabling the feature improves system performance by a variable amount, depending on the following factors:

  - How frequently new files are created
  - For indexed and relative files, how frequently existing files are extended
  - How fragmented the volume is

  Be sure to consider the security implications before you disable high-water marking.

  To disable high-water marking, you can specify the /NOHIGHWATER qualifier when initializing the volume, or you can disable high-water marking with the DCL command SET VOLUME in the following format:

  ```
  SET VOLUME/NOHIGHWATER_MARKING device-name[:]
  ```

- Set file extend parameters for OpenVMS Record Management Services (RMS)—Because files extend in increments of twice the multiblock count (default 16), system defaults provide

file extension of 32 blocks rounded up to the nearest multiple of the disk's cluster size. Thus, when files are created or extended, increased I/O may slow performance. The problem can be corrected by specifying larger values for file extend parameters or by setting the system parameter RMS_EXTEND_SIZE. See *Section 1.5, "Modifying System Parameters with AUTOGEN"* for information about modifying system parameters. Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for a description of all system parameters.

For more detailed information about establishing the file extension quantity, refer to the *Guide to Creating OpenVMS Modular Procedures*.

- Install frequently used images—When an image is accessed concurrently by more than one process on a routine basis, install the image with the Install utility (INSTALL), specifying the /OPEN, /SHARED, and /HEADER_RESIDENT qualifiers. You will thereby ensure that all processes use the same physical copy of the image, and that the image will be activated in the most efficient way.

  Generally, an image takes about two additional physical pages when installed with the /OPEN, /HEADER_RESIDENT, and /SHARED qualifiers. The utility's LIST/FULL command shows the highest number of concurrent accesses to an image installed with the /SHARED qualifier. This information can help you decide whether installing an image is an efficient use of memory.

  See *Section 3.10.11, "Installing Images with INSTALL"* and the *VSI OpenVMS System Management Utilities Reference Manual* for more information about installing images.

- On Alpha systems, install shareable and executable images specifying the /RESIDENT qualifier with the Install utility. For more information, see *Section 3.10.6, "Installing Images with Shared Address Data"*.

  Note that this is a tradeoff between the CPU and memory. Installing an image with /RESIDENT qualifier means that the code is to be nonpaged. Depending on the amount of sharing, this is can be a memory gain or loss.

- Reduce system disk I/O—You can move frequently accessed files off the system disk and use logical names to specify the location or, where necessary, other pointers to access them. For example:

  - SYSUAF.DAT (SYSUAF is the logical name)
  - RIGHTSLIST.DAT (RIGHTSLIST is the logical name)
  - VMSMAIL_PROFILE.DATA (VMSMAIL_PROFILE is the logical name)
  - NETPROXY.DAT (NETPROXY is the logical name)
  - NET$PROXY.DAT (NET$PROXY is the logical name)
  - The queue database (for more information, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*)
  - ERRFMT log files (SYS$ERRORLOG is the logical name)
  - MONITOR log files (SYS$MONITOR is the logical name)
  - The accounting log file (ACCOUNTNG is the logical name)
  - SECURITY_AUDIT.AUDIT$JOURNAL (SET AUDIT/JOURNAL=SECURITY/DESTINATION= *filespec*)

- Default DECnet for OpenVMS accounts (records included in the SYSUAF file on the OpenVMS distribution kit)

To redefine logical names for these system files, edit the site-specific command procedure SYS$MANAGER:SYLOGICALS.COM. For more information about defining logical names in SYLOGICALS.COM, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

You can also consider moving paging and swapping activity off the system disk by creating large secondary page and swap files on a less heavily used disk. However, if you want to store crash dumps for diagnosing system failures, the dump file must reside in the system-specific directory SYS$SPECIFIC:[SYSEXE] on the system disk for storing crash dumps; if no dump file exists in SYS$SPECIFIC:[SYSEXE], the primary page file must be located there if you want to store crash dumps. For detailed information about moving page and swap files, see *Section 2.15, "Creating and Modifying Page, Swap, and Dump Files"*.

- On Alpha systems, set aside large amounts of memory for use within memory-resident sections. The Reserved Memory Registry through its interface within the SYSMAN utility allows an OpenVMS Alpha system to be configured with large amounts of memory set aside for use within memory-resident sections and by other privileged applications. The Reserved Memory Registry also allows an OpenVMS system to be properly tuned through the AUTOGEN utility, taking into account the preallocated reserved memory. For more information, see *Section 3.11, "Reserved Memory Registry"*.

3.9. Expanding System Libraries

Some of the larger system libraries are shipped with the OpenVMS operating system in a data-reduced (compressed) format. Expanding (that is, decompressing) these libraries gives the system faster access to them but also consumes more disk space. To learn how to determine whether you have enough disk space to expand some or all libraries, refer to *Section 3.9.1, "Determining Disk Space Available to Expand Libraries"*.

If you have delete access to the existing libraries and write access to the library directories, you can use either of the following methods to expand and reduce (compress) libraries:

- Run the LIBDECOMP.COM command procedure (see *Section 3.9.2, "Using the Library Decompression Utility (LIBDECOMP.COM)"*), which can expand or reduce one, some, or all of the selected libraries that ship in data-reduced (compressed) format.

- Enter the DCL command LIBRARY with the /DATA qualifier (see *Section 3.9.3, "Using the LIBRARY Command with the /DATA Qualifier"*) to expand or reduce one library at a time, including libraries that do not ship in data-reduced format.

Note

In this discussion, “decompress” and “compress” refer to actions that result from using either the LIBDECOMP.COM procedure or the LIBRARY command with the /DATA=EXPAND or /DATA=REDUCE qualifier. These actions should not be confused with the results of specifying the /COMPRESS qualifier, which performs a different, unrelated function. For more information about the LIBRARY command and its qualifiers, refer to the *VSI OpenVMS Command Definition, Librarian, and Message Utilities Manual* or online help.

3.9.1. Determining Disk Space Available to Expand Libraries

Before you expand any libraries, make sure your system has enough free disk space to accommodate the expanded files. To find out how much free disk space you have, enter the following command:

```
$ SHOW DEVICE SYS$SYSDEVICE
```

The amount of space required to expand all the libraries that ship in data-reduced format differs, depending on your operating system and configuration. For Alpha systems, you need approximately 60,000 free disk blocks; VAX systems require less. For specific disk requirements, refer to the Upgrade and Installation Manual for your operating system.

If you have less free disk space than is required, or if you do not want to expand all the system libraries, you can choose to expand only some of them. For example, you might expand only the system help library (HELPLIB.HLB) and other libraries that are used frequently on your system, and leave the rest in data-reduced format.

3.9.2. Using the Library Decompression Utility (LIBDECOMP.COM)

The Library Decompression utility is a command procedure, LIBDECOMP.COM, located in the SYS $UPDATE directory. LIBDECOMP.COM uses LIBRARY commands to expand (decompress) or reduce (compress) any or all of the system libraries that ship in data-reduced format. (For a list of these libraries, see *Section 3.9.2.1, "Libraries on which LIBDECOMP.COM Operates"*.)

The LIBDECOMP.COM command procedure allows you to enter one command instead of multiple LIBRARY commands to expand or reduce all or many libraries in one operation. LIBDECOMP.COM can be executed interactively (see *Section 3.9.2.2, "Using LIBDECOMP.COM Interactively"*) or in batch mode (see *Section 3.9.2.3, "Using LIBDECOMP.COM in Batch Mode"*).

3.9.2.1. Libraries on which LIBDECOMP.COM Operates

The Library Decompression utility works on a selected set of large libraries — those that ship in data-reduced format. Some libraries are platform specific and ship only on VAX systems or only on Alpha systems. Many libraries are optional components and may not be present on your system if all options were not selected when OpenVMS was installed. *Table 3.1, "Libraries Recognized by LIBDECOMP.COM"* lists all the libraries known to LIBDECOMP.COM on both VAX and Alpha systems.

Table 3.1. Libraries Recognized by LIBDECOMP.COM

Library Name	Platform	Description
[SYSHLP] directory; help library files (.HLB)		
ACLEDT.HLB	Both	Access Control List Editor help
BKM$HELP.HLB	Both	Backup Manager help
DBG$HELP.HLB	Both	OpenVMS Debugger help
DBG$UIHELP.HLB	Both	OpenVMS Debugger help
EDTHELP.HLB	Both	EDT editor help
EVE$HELP.HLB	Both	EVE editor help
EVE$KEYHELP.HLB	Both	EVE keypad help

Library Name	Platform	Description
EXCHNGHLP.HLB	Both	Exchange utility help
HELPLIB.HLB	Both	DCL help
LANCP$HELP.HLB	Both	LAN Control Program help
LATCP$HELP.HLB	Both	LAT Control Program help
MAILHELP.HLB	Both	Mail utility help
NCPHELP.HLB	Both	Network Control Program help
SDA.HLB	Both	System Dump Analyzer help
SHWCLHELP.HLB	Both	Show Cluster utility help
SYSGEN.HLB	Both	System Generation utility help
SYSMANHELP.HLB	Both	System Management utility help
TPUHELP.HLB	Both	Text Processing Utility help
UAFHELP.HLB	Both	Authorize utility help
[SYSLIB] directory; macro library files (.MLB)		
LANIDEF.MLB	Alpha only	LAN internal driver macros
LIB.MLB	Both	Operating system macros
STARLET.MLB	Both	Operating system macros
SYSBLDMLB.MLB	VAX only	System build files
[SYSLIB] directory; object library files (.OLB)		
DECCRTL.OLB	VAX only	VSI C Run-time Library
STARLET.OLB	Both	System object library and run-time library
SYSBLDLIB.OLB	VAX only	System build files
VAXCRTL.OLB	Both	VSI C RTL routine name entry points; VAX G_floating double-precision, floating-point entry points
VAXCRTLD.OLB	Alpha only	Limited support of VAX D_floating double-precision, floating-point entry points
VAXCRTLDX.OLB	Alpha only	VAX D_floating support; support for /L_DOUBLE_SIZE=128 compiler qualifier
VAXCRTLT.OLB	Alpha only	IEEE T_floating double-precision, floating-point entry points
VAXCRTLTX.OLB	Alpha only	IEEE T_floating support; support for /L_DOUBLE_SIZE=128 compiler qualifier
VAXCRTLX.OLB	Alpha only	G_floating support; support for /L_DOUBLE_SIZE=128 compiler qualifier
VMS$VOLATILE_PRIVATE_INTERFACES.OLB	Alpha only	OpenVMS bugcheck processing codes
[SYSLIB] directory; text library files (.TLB)		
ERFLIB.TLB	Both	ANALYZE/ERROR device descriptions
LIB_ADA_SUBSET.TLB	Both	Ada programmers toolkit of operating system definitions

Library Name	Platform	Description
NTA.TLB	Both	Files to build against NTA facility
STARLET_RECENT_ ADA_SUBSET.TLB	Both	Ada programmers toolkit of operating system definitions
STARLETSD.TLB	Both	STARLET definitions used during layered product installations
SYS$LIB_C.TLB	Alpha only	Header files for C language; derived from LIB
SYS$STARLET_C.TLB	Both	Public header files for VSI C

You can use the list function of the LIBDECOMP.COM command procedure to output a list of all libraries known to the Library Decompression utility, including their size and status on your system. For details and an example, see *Section 3.9.2.2.1, "Listing Libraries"*.

3.9.2.2. Using LIBDECOMP.COM Interactively

The basic command to run the Library Decompression utility is as follows:

```
@SYS$UPDATE:LIBDECOMP [parameters]
```

The Library Decompression utility accepts up to eight optional parameters. The first parameter controls which of the utility's three functions is performed. Additional parameters control which libraries the utility processes. The three functions are described in the following sections:

- List function (see *Section 3.9.2.2.1, "Listing Libraries"*)

  Lists all VAX and Alpha libraries known to the Library Decompression utility and displays the size and status (reduced or expanded) of libraries on your system.

- Expand function (see *Section 3.9.2.2.2, "Expanding (Decompressing) Libraries"*)

  Expands libraries that are in data-reduced format. This is the default function.

- Reduce function (see *Section 3.9.2.2.3, "Reducing (Compressing) Libraries"*)

  Compresses expanded libraries into data-reduced format.

To get a brief online help display, enter the following command:

```
$ @SYS$UPDATE:LIBDECOMP HELP
```

3.9.2.2.1. Listing Libraries

To list all VAX and Alpha libraries known to the Library Decompression utility, along with their size and status on your system, specify the LIST parameter in the command, as follows:

```
$ @SYS$UPDATE:LIBDECOMP LIST
```

The resulting list indicates which libraries are not present on your system, either because they do not ship with VAX or with Alpha or because the facility associated with them is not installed on your system. For libraries that are present, the list includes the library size and the current status (reduced or expanded). The following example shows the output from an Alpha system. (Note: File sizes are subject to change. For the most accurate information, refer to the LIST output on your own system.)

```
 $ @SYS$UPDATE:LIBDECOMP LIST
                   OpenVMS Library Decompression Utility
```

```
                List of all libraries known to LIBDECOMP
       "Library not present" indicates not installed on this system
 Libraries in SYS$SYSROOT:
       Library                      Size
   1) [SYSHLP]ACLEDT.HLB             70   Reduced format
   2) [SYSHLP]BKM$HELP.HLB          156   Reduced format
   3) [SYSHLP]DBG$HELP.HLB         1234   Reduced format
   4) [SYSHLP]DBG$UIHELP.HLB        269   Reduced format
   5) [SYSHLP]EDTHELP.HLB           154   Reduced format
   6) [SYSHLP]EVE$HELP.HLB          676   Reduced format
   7) [SYSHLP]EVE$KEYHELP.HLB        99   Reduced format
   8) [SYSHLP]EXCHNGHLP.HLB          83   Reduced format
   9) [SYSHLP]HELPLIB.HLB          9179   Reduced format
  10) [SYSHLP]LANCP$HELP.HLB        119   Reduced format
  11) [SYSHLP]LATCP$HELP.HLB        157   Reduced format
  12) [SYSHLP]MAILHELP.HLB          211   Reduced format
  13) [SYSHLP]NCPHELP.HLB           261   Reduced format
  14) [SYSHLP]SDA.HLB               308   Reduced format
  15) [SYSHLP]SHWCLHELP.HLB         103   Reduced format
  16) [SYSHLP]SYSGEN.HLB            337   Reduced format
  17) [SYSHLP]SYSMANHELP.HLB        492   Reduced format
  18) [SYSHLP]TPUHELP.HLB           575   Reduced format
  19) [SYSHLP]UAFHELP.HLB           241   Reduced format
  20) [SYSLIB]LANIDEF.MLB           181   Reduced format
  21) [SYSLIB]LIB.MLB              2715   Reduced format
  22) [SYSLIB]STARLET.MLB          2335   Reduced format
  23) [SYSLIB]SYSBLDMLB.MLB                        Library not present
  24) [SYSLIB]DECCRTL.OLB                          Library not present
  25) [SYSLIB]STARLET.OLB         27461   Reduced format
  26) [SYSLIB]SYSBLDLIB.OLB                        Library not present
  27) [SYSLIB]VAXCRTL.OLB          1163   Reduced format
  28) [SYSLIB]VAXCRTLD.OLB         1587   Reduced format
  29) [SYSLIB]VAXCRTLDX.OLB        1506   Reduced format
  30) [SYSLIB]VAXCRTLT.OLB         1434   Reduced format
  31) [SYSLIB]VAXCRTLTX.OLB        1449   Reduced format
  32) [SYSLIB]VAXCRTLX.OLB         1285   Reduced format
  33) [SYSLIB]ERFLIB.TLB             64   Reduced format
  34) [SYSLIB]LIB_ADA_SUBSET.TLB   1839   Reduced format
  35) [SYSLIB]NTA.TLB                34   Reduced format
  36) [SYSLIB]STARLETSD.TLB        3940   Reduced format
  37) [SYSLIB]SYS$LIB_C.TLB        9442   Reduced format
  38) [SYSLIB]SYS$STARLET_C.TLB    5864   Reduced format
  39) [SYSLIB]VMS$VOLATILE_PRIVATE_INTERFACES.OLB   445   Reduced format
  40) [SYSLIB]STARLET_RECENT_ADA_SUBSET.TLB   1100   Reduced format
  Total Libraries:  37           78568
```

3.9.2.2.2. Expanding (Decompressing) Libraries

The default action for LIBDECOMP.COM is the expand function. You can explicitly specify EXPAND as the first parameter on your command line, but this is not necessary. Unless the first parameter is LIST or REDUCE, the expand function is used by default.

If the expand function is used, the remaining parameters specify which of the libraries to expand. You can specify ALL to expand all libraries known to the utility, or you can specify up to eight library names (seven names if EXPAND is specified). Wildcard characters are not permitted. The specified libraries must be known to the utility. (To expand any other libraries, you must use the LIBRARY command, as described in *Section 3.9.3, "Using the LIBRARY Command with the /DATA Qualifier"*.) If you do not

specify ALL or a list of libraries, LIBDECOMP.COM prompts you for the libraries to expand. You can specify any number of libraries in response to the prompt.

Note

Expanding or reducing all of the libraries known to LIBDECOMP.COM will generally take about five to ten minutes. However, depending on the specific hardware and software configuration of your system, and on other activity, this can take longer, up to half an hour or more in some cases.

The expand function produces a display similar to the one shown in the following OpenVMS Alpha example. After the header lines are displayed, there is a pause while LIBDECOMP.COM checks the status of each library.

```
              OpenVMS Library Decompression Utility
                Candidate Libraries to be expanded
 (Libraries not present and libraries already expanded are not listed)
   1  ACLEDT.HLB           13  NCPHELP.HLB          25  VAXCRTLD.OLB
   2  BKM$HELP.HLB         14  SDA.HLB              26  VAXCRTLDX.OLB
   3  DBG$HELP.HLB         15  SHWCLHELP.HLB        27  VAXCRTLT.OLB
   4  DBG$UIHELP.HLB       16  SYSGEN.HLB           28  VAXCRTLTX.OLB
   5  EDTHELP.HLB          17  SYSMANHELP.HLB       29  VAXCRTLX.OLB
   6  EVE$HELP.HLB         18  TPUHELP.HLB          30  ERFLIB.TLB
   7  EVE$KEYHELP.HLB      19  UAFHELP.HLB          31  LIB_ADA_SUBSET.TLB
   8  EXCHNGHLP.HLB        20  LANIDEF.MLB          32  NTA.TLB
   9  HELPLIB.HLB          21  LIB.MLB              33  STARLETSD.TLB
  10  LANCP$HELP.HLB       22  STARLET.MLB          34  SYS$LIB_C.TLB
  11  LATCP$HELP.HLB       23  STARLET.OLB          35  SYS$STARLET_C.TLB
  12  MAILHELP.HLB         24  VAXCRTL.OLB
       36  VMS$VOLATILE_PRIVATE_INTERFACES.OLB
       37  STARLET_RECENT_ADA_SUBSET.TLB

           A  ALL libraries to be expanded
           H  Display HELP information for LIBDECOMP
           E  EXIT this procedure
```

If you specified ALL, the following message is displayed, and the utility then expands all listed libraries.

```
"ALL" specified; all libraries will be processed
```

If you did not specify ALL, you are prompted as follows:

```
* Enter a letter or the number(s) of libraries to be expanded
  (Separate multiple numbers with a comma)
```

Enter A, H, E, or one or more numbers to indicate which libraries to expand. There is no limit to how many numbers you can specify.

If you enter parameters to identify specific libraries to expand, LIBDECOMP.COM does not display the list as shown in the example; rather, it lists each library as it is processed.

Examples

- Either of the following commands expands all libraries:

```
$ @SYS$UPDATE:LIBDECOMP ALL
$ @SYS$UPDATE:LIBDECOMP EXPAND ALL
```

- To choose whether to expand some or all of the libraries in a menu, first enter one of the following commands:

  ```
  $ @SYS$UPDATE:LIBDECOMP
  $ @SYS$UPDATE:LIBDECOMP EXPAND
  ```

  Then, as shown in an earlier example, you are prompted to specify all libraries or to specify the numbers for an unlimited number of individual libraries to be expanded.

- To bypass the menu and expand only selected libraries, enter a command similar to one of the following, in which you specify the library names in the command line instead of responding to a prompt:

  ```
  $ @SYS$UPDATE:LIBDECOMP HELPLIB.HLB STARLET.MLB LIB.MLB
  $ @SYS$UPDATE:LIBDECOMP EXPAND HELPLIB STARLET.MLB LIB
  ```

  Both commands have the same result: to expand the HELPLIB.HLB, STARLET.MLB, and LIB.MLB libraries.

3.9.2.2.3. Reducing (Compressing) Libraries

When you specify REDUCE as the first parameter in your command line, LIBDECOMP.COM reduces libraries that have been expanded.

You can specify ALL after REDUCE to reduce all of the libraries known to the utility, or you can specify a list of up to seven libraries. Wildcard characters are not permitted.

If you do not specify ALL or the name of at least one library, LIBDECOMP.COM prompts you for the libraries to reduce. There is no limit to the number of libraries you can list in response to the prompt.

The reduce function produces a display similar to that of the expand function except that it displays only the libraries eligible to be reduced.

Examples

- The following command reduces all libraries that have been expanded:

  ```
  $ @SYS$UPDATE:LIBDECOMP REDUCE ALL
  ```

- To choose whether to reduce some or all of the libraries in a menu, first enter the following command:

  ```
  $ @SYS$UPDATE:LIBDECOMP REDUCE
  ```

  Then, as with the expand function, you are prompted to specify all libraries or to specify the numbers for an unlimited number of individual libraries to be reduced.

3.9.2.3. Using LIBDECOMP.COM in Batch Mode

You can submit the Library Decompression utility to a batch queue by using the DCL command SUBMIT with the /PARAMETERS qualifier, as follows:

```
SUBMIT SYS$UPDATE:LIBDECOMP /PARAMETERS=(p1[,p2,...])
```

The batch procedure produces the same results as the interactive procedure, but with the batch job you must specify HELP, LIST, ALL, or at least one library name, because the batch job cannot prompt you for input.

You can specify up to eight parameters. If you specify more than one parameter, the parameters must be enclosed in parentheses and must be separated by commas.

Examples

- The following command outputs a list of all the libraries known to LIBDECOMP.COM:

```
$ SUBMIT SYS$UPDATE:LIBDECOMP /PARAMETERS=LIST
```

- By default, the following command expands all eligible libraries:

```
$ SUBMIT SYS$UPDATE:LIBDECOMP /PARAMETERS=ALL
```

- The following command expands the HELPLIB.HLB, STARLET.MLB, and LIB.MLB libraries:

```
$ SUBMIT SYS$UPDATE:LIBDECOMP -
_$ /PARAMETERS=(EXPAND,HELPLIB.HLB,STARLET.MLB,LIB.MLB)
```

3.9.3. Using the LIBRARY Command with the /DATA Qualifier

An alternative to using the Library Decompression utility to expand or reduce an individual library is to use the DCL command LIBRARY with the /DATA qualifier. With this method, you can specify only one library per LIBRARY command.

Unlike LIBDECOMP.COM, which acts on only about 40 selected libraries that ship in data-reduced format, the LIBRARY command can be used to expand or reduce almost *any* library file, provided the library type (the file extension) is known to the OpenVMS Librarian utility. System libraries that you should not compress include the following:

- [SYSLIB]EPC$FACILITY.TLB

  This file ships empty.

- [SYSLIB]IMAGELIB.OLB

  Compression is not effective on this library.

- [SYSLIB]NCS$LIBRARY.NLB

  Library type .NLB is not known to the OpenVMS Librarian utility.

To expand a specified library, use the following command format:

```
LIBRARY library-name.ext /DATA=EXPAND
```

To reduce a specified library, use the following command format:

```
LIBRARY library-name.ext /DATA=REDUCE
```

You must always specify the library extension (.HLB, .MLB, .OLB, or .TLB).

If the specified library is not located on your current default device and directory, you must also specify the device and directory in the library specification. Most system libraries are in either SYS$HELP ([SYSHLP]) or SYS$LIBRARY ([SYSLIB]). For example:

```
$ LIBRARY [SYSHLP]HELPLIB.HLB /DATA=EXPAND
```

For information about other qualifiers for the LIBRARY command, see online help for LIBRARY or refer to the *VSI OpenVMS Command Definition, Librarian, and Message Utilities Manual*.

3.10. Using INSTALL to Install Known Images

The Install utility (INSTALL) stores information about images in memory. Use INSTALL for the following reasons:

Reason	**For More Information**
To conserve memory use for images that are used concurrently	*Section 3.10.7, "Installing Images to Conserve Physical Memory"*
To improve system performance	*Section 3.10.5, "Installing Images to Improve Image Activation Performance"*
On Alpha systems, to improve performance by using images with shared address data (Alpha specific)	*Section 3.10.6, "Installing Images with Shared Address Data"*
To make executable images that require enhanced privileges available for general use	*Section 3.10.9, "Activating Images in a Privileged Context"*
To allow a nonprivileged image to call the privileged functions of a shareable image	*Section 3.10.9, "Activating Images in a Privileged Context"*
To mark a sharable image as trusted so it can be invoked by privileged executable images	*Section 3.10.9, "Activating Images in a Privileged Context"*

The site-independent startup command procedure, STARTUP.COM, uses INSTALL to install certain system images when the system boots. You use INSTALL to install other selected images, according to the needs of your site.

Installed images must be reinstalled each time the system reboots. To do so, include INSTALL commands in the site-specific startup command procedure SYSTARTUP_VMS.COM, as explained in *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

The Install utility (INSTALL) only installs images that are linked with the /NOTRACEBACK qualifier.

Note that INSTALL commands perform a different function than System Generation utility (SYSGEN) INSTALL commands.

The following sections explain installed images and how to use the Install utility.

3.10.1. Understanding Images and Known Images

An **image** is a collection of procedures and data bound together by the Linker utility. Executable images can be executed (or run) in a process, either by a command line interpreter (CLI) or the $CREPRC system service. Usually, executable programs have the file type .EXE.

There are three types of images:

Image Type	Description
Executable	An image linked with the /EXECUTABLE qualifier (or without the /SHAREABLE qualifier) of the Linker utility. For more information, refer to the *VSI OpenVMS Linker Utility Manual*.
Shareable	An image linked with the /SHAREABLE qualifier of the Linker utility. Shareable images are sometimes referred to as linkable images because they can be specified – implicitly or explicitly – as input files to the link of another file. A shareable image is not copied into the executable images that link with it. Thus, only one copy of the shareable image needs to be on disk, no matter how many executable images have linked with it. For more information, refer to the *VSI OpenVMS Linker Utility Manual*.
System	An image that does not run under the control of the operating system. It is intended for standalone operation only. The content and format of a system image differs from that of shareable images and executable images. For more information, refer to the *VSI OpenVMS Linker Utility Manual*.

When you install an image with INSTALL, the image is assigned attributes and becomes known to the system. For this reason, an installed image is also called a **known image**.

The image activator processes search lists in two passes, in order to favor known images. On its first pass through the search list, the image activator looks up images as known files. If needed, on a second pass through the search list, the image activator looks up images on disk.

3.10.2. Understanding Known File Entries

The system defines known images in internal data structures called **known file entries**. Each entry identifies the file name of the installed image and the attributes with which it was installed (for information about attributes of installed images, see *Section 3.10.3, "Understanding Attributes You Can Assign to Known Images"*).

Known file entries last only while the system is operating. If the system is shut down or fails for any reason, you must reinstall all known images after the system is rebooted.

3.10.3. Understanding Attributes You Can Assign to Known Images

By specifying appropriate qualifiers to INSTALL commands, you can assign attributes to known images. *Table 3.2, "Attributes of Known Images"* describes these attributes and the qualifiers that are used to assign them to known images.

Table 3.2. Attributes of Known Images

Attribute	Description	Qualifier
Header resident	The header of the image file (native images only) remains permanently resident, saving one disk I/O	/[NO] HEADER_RESIDENT

Attribute	Description	Qualifier
	operation per file access. For images with single-block file headers, the cost is less than 512 bytes of paged dynamic memory per file; for images with multiblock headers, the cost varies according to the header block count. Images installed header resident are implicitly installed permanently open.	
Permanently open	The image file remains open, so access to the image does not require a call to the file system.	/OPEN
Privileged	Amplified privileges are temporarily assigned to any process running the image, permitting the process to exceed its user authorization file (UAF) privilege restrictions during execution of the image. In this way, users with normal privileges can run programs that require higher-than-normal privileges. This attribute (and the /PRIVILEGED qualifier that creates it) applies only to executable images.	/PRIVILEGED[=(priv-name[,...])]
Protected	When the image is activated, the address space for the image is protected against modification by user-mode code. This is critical for shareable code that runs in kernel or executive mode.	/PROTECTED
Resident(Alpha specific)	On Alpha systems, code or read-only data for an image is made permanently resident in a system region of memory. This improves performance by using a special page mapping to reduce translation buffer (TB) miss rates. The resident attribute applies to shareable or executable images that have been linked with the qualifier	/SECTION_BINDING=(CODE,DATA).
Shared	More than one user can access the read-only and non-copy-on-reference read/write sections of the image concurrently, so that only one copy of those sections needs to be in physical memory. (Copy-on-reference sections always require a separate copy for each process.) The image is implicitly declared permanently open.	/SHARED
Writable	When a shareable non-copy-on-reference writable section is removed from physical memory (for paging reasons or because no processes are referencing it), it is written back to the image file. Any updates made by processes mapped to the section, therefore, are preserved (while the initial values are lost). The image must also be declared shareable.	/WRITABLE

3.10.4. Determining Which Images to Install

You can install images for the following reasons:

- Improve image activation performance
- On Alpha systems, improve execution performance of shared images

- Conserve physical memory
- Enhance privileges of images during execution

Because an installed file requires system resources, such as paged dynamic memory, install those files that most improve system performance and site requirements. The INSTALL command LIST provides information about installed images to help you evaluate the merits of installing images. For example, the LIST command calculates the number of times each image is accessed, and shows the number of concurrent accesses, so you can determine if the installation of the images is worth the overhead.

3.10.5. Installing Images to Improve Image Activation Performance

You can improve image activation performance by installing images that run frequently. Image activation performance improves when programs are installed because the operating system opens installed files by file ID rather than by file name, thus eliminating costly directory operations.

Installing images as header resident further enhances activation performance because the system avoids the overhead of I/O operations to read the image header into memory.

To install an image as header resident, specify the /HEADER_RESIDENT qualifier when you install the image. This makes the header of the image file remain permanently resident, saving disk I/O. Specifying the /HEADER_RESIDENT qualifier implicitly makes the images permanently open.

Image headers are stored in paged dynamic memory. The size of the image headers varies.

Frequently accessed images, critical to a site's operations, can be installed as open images. To install an image as permanently open, specify the /OPEN qualifier when you install the image. The image file remains open, so access to the image does not require a call to the file system. The cost of keeping an image file permanently open is approximately 512 bytes of nonpaged dynamic memory per file.

3.10.6. Installing Images with Shared Address Data

Using shared address data on OpenVMS Alpha systems improves performance at the following times:

- At run time, shared address data saves physical memory because of increased memory sharing between processes.
- At image activation, shared address data reduces CPU and I/O time because fixup is performed at installation time.

For details, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

Related Terms

Explanations of terms related to shared address data follow.

- Image section

  An image consists of a number of image sections. An image section can contain:

  - Instructions (code)
  - Read-only data (constants)
  - Read/write data

- Shared known images

  You can make an image a shared known image by using the following command:

  ```
  $ INSTALL ADD image-name /SHARED
  ```

  When you enter this command, the Install utility creates global sections for read-only image sections, allowing the sections to be shared by all the processes that run the image.

- Address data

  One kind of image section contains address data. At execution time, address data sections are read-only. However, the addresses are not known until image activation; therefore, the image section is read/write until the end of image activation. Addresses for a shareable image generally vary with the process because different processes are likely to have different collections of mapped images.

- Shared address data

  The shared address data feature assigns unique P1 space addresses for shareable images from the P1 image region. (The IMGREG_PAGES system parameter determines the size of the P1 space.) With the assigned address, the Install utility determines the content of an address data section when the image is installed. A global section is created to allow shared access to each address data image section.

  Executable (main) images can also use shared address data sections; these images are not assigned P1 addresses, however, because the base address for an executable image is determined when the image is linked.

3.10.6.1. System-Provided Images

Many images that are part of the OpenVMS software product are installed as shared known images with shared address data. This provides the performance benefit without requiring the system manager to take any explicit action.

3.10.6.2. Application Images

As system manager, you might choose to install additional images with shared address data. In considering this option, you need to investigate application dependencies on sharable images.

3.10.7. Installing Images to Conserve Physical Memory

You can conserve physical memory by installing images that usually run concurrently from several processes. When an image is not installed, or is installed without the shared attribute, each process running the image requires private sections in memory. Shared images conserve physical memory because only one copy of the code needs to be in memory at any time, and many users can access the code concurrently. Use the /SHARED qualifier to install images as shared images.

When you install an image with the shared attribute, permanent system global sections are created. Execution of non-copy-on-reference global sections requires only one copy per section to be in physical memory, no matter how many processes are running the image to which the sections belong.

The number of images you can install with the shared attribute is restricted by the GBLPAGES and GBLSECTIONS system parameters. For more information about these system parameters, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

3.10.8. Installing Images to Enhance Privileges of Images

There are two ways to allow an image to execute in an enhanced privilege environment:

- Installing executable images with privileges to allow a nonprivileged process to perform the privileged functions of the image.

  Use the /PRIVILEGED qualifier with the INSTALL commands CREATE or REPLACE.

  Caution

  Installing an image with enhanced privilege can compromise system security. Make sure the image does not enable a user to regain control with extra privileges enabled.

- Installing protected shareable images (which are used to implement user-written system services), allowing other, nonprivileged images to execute select portions of privileged code without enhancing the privileges of those individual images.

  Use the /PROTECTED and /SHARED qualifiers with the INSTALL commands CREATE or REPLACE.

3.10.8.1. Privileged Executable Images

A nonprivileged process can perform the privileged functions of an executable image when the image is installed with privileges. Install executable images with enhanced privileges by using the /PRIVILEGED qualifier; amplified privileges are temporarily assigned to any process running the image (executable images only), permitting the process to exceed its user authorization file (UAF) privilege restrictions during execution of the image. In this way, users with normal privileges can run programs that require higher-than-normal privileges.

For an image installed with privileges to activate another image, such as a shareable image, either by having it linked to the privileged image or by using LIB$FIND_IMAGE_SYMBOL, the following conditions hold:

- The shareable image must be installed as a known image using INSTALL.
- Logical names and table names used to find the image must be defined in executive or kernel mode. In particular, the standard executive-mode definition of LNM$FILE_DEV translates only to LNM $SYSTEM; definitions in the process, job, or group tables are not recognized.
- Only images linked with the Linker utility qualifiers /NODEBUG and /NOTRACE can be installed with enhanced privilege.

3.10.8.2. Privileged Shareable Images

A privileged shareable image is a shareable image with defined entry points that execute in inner (executive or kernel) mode. Inner-mode entry points in shareable images are referred to as user-written system services.

To create a privileged shareable image, you must:

1. Create a program section with the VECTOR attribute containing a PLV (privileged library vector) data structure.

2. Link the shareable image with the /PROTECT command qualifier or the PROTECT= option of the Linker utility, so that the image acquires its particular form of enhanced privileges.

   - Use the /PROTECT command qualifier when all parts of an image require protection.
   - Use the PROTECT= option when only part of a privileged shareable image requires protection.

3. Install the privileged shareable image with the Install utility, specifying both the /PROTECTED and the /SHARED qualifiers. The /PROTECTED qualifier assigns the protected attribute. The /SHARED qualifier assigns the shareable attribute. See *Section 3.10.3, "Understanding Attributes You Can Assign to Known Images"* for information about these attributes.

Note

You cannot grant privileges to a shareable image using the /PRIVILEGED qualifier for the INSTALL commands ADD or CREATE. This qualifier works only for executable images.

For more information about creating privileged shareable images, refer to the *VSI OpenVMS Programming Concepts Manual*.

3.10.9. Activating Images in a Privileged Context

When a process performs one of the following actions, the image activator enters a restricted mode of operation similar to that entered when a privileged program is run:

- Runs an executable or shareable image to which it has execute but not read access.
- Activates an executable image installed with the /PRIVILEGE or /EXECUTE_ONLY qualifier.
- Activates an image called by a privileged shareable image.

In this mode of operation:

- All shareable images activated during the life of the execute-only image must be installed.
- The image activator directs OpenVMS RMS to use only **trusted logical names** (logical names associated with executive or kernel mode) when opening image files.

  Note

  The executable image that calls an execute-only shareable image must be installed with the /EXECUTE_ONLY qualifier, which enables the executable image to activate shareable images to which the process has execute but not read access.

  The /EXECUTE_ONLY qualifier has meaning only for executable images.

  This restriction assures that shareable images running in a privileged context can be trusted to behave as expected.

3.10.10. Specifying File Names in INSTALL

When you use INSTALL commands, your file specifications must name existing executable or shareable images. OpenVMS RMS resolves each file specification using the following defaults:

- A device and directory type of SYS$SYSTEM
- A file type of .EXE

You can specify a specific version of the file as the known version of the image with the CREATE or REPLACE command. Even if other versions of the file exist, the version that you specify will be the version that satisfies all known file lookups for the image.

3.10.11. Installing Images with INSTALL

Before performing this task, you should understand the following points:

- Attributes of installed images. For information, see *Section 3.10.3, "Understanding Attributes You Can Assign to Known Images"*.
- File specifications for the Install utility. For information, see *Section 3.10.10, "Specifying File Names in INSTALL"*.

How to Perform This Task

1. Give yourself the CMKRNL privilege by entering the following command:

   ```
   $ SET PROCESS/PRIVILEGES=CMKRNL
   ```

2. Invoke INSTALL by entering the following command:

   ```
   $ INSTALL
   ```

3. Enter the CREATE command in the following format:

   ```
   CREATE file-spec [/qualifier...]
   ```

   Specify one or more of the following qualifiers, depending on which attributes you want to assign to the image:

   /EXECUTE_ONLY
   /HEADER_RESIDENT
   /OPEN
   /PRIVILEGED
   /PROTECTED
   /RESIDENT (Alpha systems only)
   /SHARED
   /WRITABLE

   For more information about installing images, refer to the INSTALL command CREATE in the *VSI OpenVMS System Management Utilities Reference Manual*.

   ### Note

   Installing the Install utility itself requires that a number of shareable images have been previously installed. If any of those required shareable images (such as SMG$SHR, LIBOTS, and so on) is unavailable, the execution of the Install utility fails. Since INSTALL will not work in this situation, you cannot simply install the missing images. To work around this problem, redefine the INSTALL command as follows:

   ```
   $ DEFINE INSTALL SYS$SYSTEM:INSTALL.EXE;0
   ```

When you now enter the INSTALL command, the image activator does not check the known files list for INSTALL.EXE, and the INSTALL command will complete, allowing you to install the required shareable images.

3.10.12. Displaying Known Images with INSTALL

Use the INSTALL command LIST to display information about known images.

The information displayed with the /FULL qualifier of the LIST command can help you determine if installing an image is worth the expense.

How to Perform This Task

1. Invoke INSTALL by entering the following command:

   ```
   $ INSTALL
   ```

2. To display a list of all known images and their attributes, enter the LIST command. To display attributes for a specific image, specify the name of the image as follows:

   ```
   LIST file-spec
   ```

   For example:

   ```
   INSTALL> LIST LOGINOUT
   ```

3. To display complete information about a specific image, including the number of accesses, the number of concurrent accesses, and the number of global sections created, specify the /FULL qualifier as follows:

   ```
   LIST/FULL file-spec
   ```

Example

The following example displays complete information about the installed image LOGINOUT.EXE, including the number of accesses, the number of concurrent accesses, and the number of global sections created:

```
$ INSTALL
INSTALL> LIST/FULL LOGINOUT
DISK$VMS551:<SYS2.SYSCOMMON.SYSEXE>.EXE
   LOGINOUT;2       Open Hdr Shar Prv
        Entry access count         = 36366
        Current / Maximum shared   = 1 / 10
        Global section count       = 3
        Privileges = CMKRNL SYSNAM LOG_IO ALTPRI TMPMBX SYSPRV
INSTALL>
```

3.10.13. Defining Logical Names for Shareable Image Files

If a shareable image is not located in SYS$SHARE, you must define a logical name for that image in order to run an executable image linked against it. For example, if the file specification for STATSHR is SYS$SHARE:STATSHR.EXE, no logical name is necessary. But if you put STATSHR in SYS

$DEVICE:[TEST], you must define STATSHR as a logical name before running an executable image that calls it. The logical name must be the same one that was used as the input file specification for the shareable image when it was linked (this is the same name used in installation). For example:

```
$ DEFINE STATSHR SYS$SYSDEVICE:[TEST]STATSHR
```

By redefining the logical name of a shareable image, you can replace that shareable image with another without requiring the calling executable image to relink. For example, the following statement redefines the file name STATSHR. It becomes the logical name of the shareable image SYS$SYSDEVICE:[MAIN]STATSHR.EXE for executable images calling STATSHR.

```
$ DEFINE STATSHR SYS$SYSDEVICE:[MAIN]STATSHR
```

Note

Logical names defined in the process or group logical name table are ignored when you run a privileged executable image. Only logical names and table names defined in executive or kernel modes are used to find the image.

3.10.14. Removing Known Images

The INSTALL command REMOVE removes a known file entry for an image and deletes any global sections created when the image was installed. Note that a volume cannot be dismounted while any known file entries are associated with it. To dismount a volume, you must delete all known images associated with it. You must also wait for all processes using those images to exit. Use the DCL command SHOW DEVICES/FILES to determine the status of the files.

For more information about the INSTALL command DELETE, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

3.11. Reserved Memory Registry

The Reserved Memory Registry through its interface within the SYSMAN utility allows an OpenVMS Alpha system to be configured with large amounts of memory set aside for use within memory-resident sections and by other privileged applications. The Reserved Memory Registry also allows an OpenVMS system to be properly tuned through the AUTOGEN utility, taking into account the preallocated reserved memory.

With the Reserved Memory Registry you can:

- Reserve system nonfluid memory for the memory-resident global section fault option.
- Reserve suitable physical pages as well as system non-fluid memory for the memory-resident global section allocate option.

The Reserved Memory Registry includes the ability to specify that the preallocated pages are to be zeroed during the booting of the system. This option reduces the time required to create the memory-resident global demand-zero section.

Another option within the Reserved Memory Registry is to include the size of the page tables required to map to the memory-resident global section in the reserved memory. If this option is specified and the reserved memory is being used for a memory-resident global section, the memory-resident global section is created with shared page tables.

3.11.1. Using the Reserved Memory Registry

OpenVMS provides a mechanism to reserve non-fluid memory for use within a memory-resident global demand-zero section. The reserved memory may either be simply a deduction from the system's nonfluid memory size or pages may be preallocated.

Using the Reserved Memory Registry ensures that AUTOGEN tunes the system properly to not include memory-resident section pages in its calculation of the system's fluid page count. AUTOGEN sizes the system pagefile, number of processes and working set maximum size based on the system's fluid page count. A system can experience severe performance problems if AUTOGEN adjusts parameters based upon a fluid page count that does not account for the physical memory that is permanently reserved for some other purpose.

Using the Reserved Memory Registry also ensures that contiguous, aligned memory is available for memory-resident sections when the allocate option is used.

Note

Although this section describes how to use the reserved memory registry for global sections, this feature can be used for other privileged applications.

3.11.1.1. Reserved Memory Registry Data File

Consumers of reserved, nonfluid memory enter the characteristics of the memory into a data file that is read during the system initialization (boot-time). The mechanics of manipulating the data file are similar to SYS$LOADABLE_IMAGES:VMS$SYSTEM_IMAGES.DATA (indicates installation-specific executive loaded images).

This file is called:

```
SYS$SYSTEM:VMS$RESERVED_MEMORY.DATA
```

The file is maintained by the SYSMAN utility (as is the executive loaded image data file).

3.11.1.2. AUTOGEN

The Reserved Memory Registry file, VMS$RESERVED_MEMORY.DATA, is read by the AUTOGEN feedback mechanism and factors into the setting of the system's fluid page count. AUTOGEN sizes the system pagefile, number of processes and working set maximum size based on the system's fluid page count.

3.11.1.3. Adding Entries to the Reserved Memory Registry

You add an entry to the data file by using the SYSMAN utility. The SYSMAN command is as follows:

```
SYSMAN RESERVED_MEMORY ADD gs_name -
                       /GROUP = n -
                       /SIZE = {size of reserved memory, unit: MB} -
                       /[NO]ALLOCATE -
                       /[NO]ZERO -
                       /[NO]PAGE_TABLES
```

- The gs_name field is the name of the memory-resident global section associated with this reserved memory. A name must be specified.

- If the /GROUP qualifier is not present, the reserved memory is for a system global section (SYSGBL).
- If the /GROUP qualifier is present, the reserved memory is for a group global section. The value UIC group number (in octal) of the process creates the group global section. Only processes within the creator's UIC group number are allowed access to the global section. For example, if a process with the UIC of [6,100] is the creator of the group global section, the group number specified for the /GROUP qualifier would be 6.
- If the /ALLOCATE qualifier is not specified or if the /NOALLOCATE qualifier is specified, the reserved memory is not allocated during the next reboot of the system. The reserved memory is only deducted from the system's fluid page count and the creation of the memory-resident global section results in the fault option being used.
- If the /ALLOCATE qualifier is specified, pages are allocated during the next reboot of the system. The allocated memory is deducted from the system's fluid page count and the creation of the memory-resident global section results in the allocate option being used. The physical alignment of the pages is based on the maximum granularity hint factor that can be used to map the pages given the size of the reserved memory. Possible granularity hint factors are 512 pages (or 4 MB) and 64 pages (or 512 KB). Therefore, assuming an 8-KB system page size, reserved memory is physically aligned as follows:

  1. `size >= 4 MB: physically aligned on a 4 Mbyte boundary`
  2. `size < 4 MB: physically aligned on a 512 KB boundary`

- If the /ZERO qualifier is not specified or if /NOZERO is specified, the preallocated pages are not zeroed during system initialization. The pages are zeroed at the time the global section is created.
- The /ZERO qualifier is only allowed if the /ALLOCATE qualifier is specified. If the /ZERO qualifier is specified, the preallocated pages are zeroed during system initialization. Zeroed pages are required for memory-resident global sections; however, the pages need not be zeroed during system initialization.
- If the /PAGE_TABLES qualifier is not specified or if /NOPAGE_TABLES is specified, additional memory is not reserved for shared page tables. When the memory-resident global section is created, shared page tables are not created for the global section.
- If the /PAGE_TABLES qualifier is specified, additional memory is reserved for shared page tables. When the memory-resident global section is created, shared page tables are created for the global section. If the /ALLOCATE qualifier is not specified or if /NOALLOCATE is specified, the additional reserved memory is only deducted from the system's fluid page count. If the /ALLOCATE qualifier is specified, additional pages are allocated during the next reboot of the system for the shared page tables and the additional reserved memory is deducted from the system's fluid page count.

3.11.2. Removing Entries from the Reserved Memory Registry

You can remove a reserved memory entry by issuing the following SYSMAN command:

```
SYSMAN RESERVED_MEMORY REMOVE gs_name /GROUP = n
```

The specified gs_name is the name of the memory-resident section associated with the entry being removed from the Reserved Memory Registry. A name must be specified.

The value n specified by the /GROUP qualifier is the UIC group number (in octal) associated with the memory-resident section being removed. If the memory-resident global section is a group section, you must specify the /GROUP qualifier. If the memory-resident global section is a system global section, you must not specify the /GROUP qualifier.

If page tables are reserved for the named memory-resident global section, the additional reserved memory is also removed.

The REMOVE command only removes entries from the Reserved Memory Registry data file; it does not affect memory within the running system.

3.11.2.1. Allocating Reserved Memory

During system initialization, the VMS$RESERVED_MEMORY.DATA data file is read.

For each entry in the data file, the number of megabytes is deducted from the system's fluid page count for this memory-resident global section as specified by the /SIZE qualifier on the RESERVED_MEMORY ADD command. If /PAGE_TABLES was specified, the amount of memory required for the shared page tables mapping the memory-resident global section is deducted from the system's fluid page count as well.

If /ALLOCATE was specified on the RESERVED_MEMORY ADD command, a suitable chunk of physical pages is also allocated and set aside for the memory-resident global section. If /PAGE_TABLES was specified, an additional chunk of physical pages is allocated and set aside for the shared page tables. The pages have a physical alignment appropriate for the largest granularity hint factor for the given sized chunk. If /ZERO was specified, the pages are zeroed during system initialization or when the system is idle. If /ZERO was not specified or if /NOZERO was specified, the pages are zeroed at the time the memory-resident global section is created.

If the system parameter STARTUP_P1 is set to MIN, entries in the Reserved Memory Registry entries are ignored and memory is not reserved.

If errors are encountered during system initialization while processing the Reserved Memory Registry data file, with reserving system fluid pages, or with allocating contiguous, aligned physical pages, an error message is issued to the console and the system continues to boot.

3.11.2.2. Freeing Reserved Memory

In the running system, you can free reserved memory by issuing the following SYSMAN command:

```
SYSMAN> RESERVED_MEMORY FREE gs_name /GROUP = n
```

The specified gs_name is the name of the memory-resident section associated with the entry being freed from the Reserved Memory Registry. A name must be specified.

The value n specified by the /GROUP qualifier is the UIC group number (in octal) associated with the memory-resident section being freed. If the memory-resident global section is a group global section, you must specify the /GROUP qualifier. If the memory-resident global section is a system global section, you must not specify the /GROUP qualifier.

If physical pages were not preallocated during system initialization for this global section, the reserved memory is simply added to the system's fluid page count. Otherwise the physical pages are deallocated onto the system's free or zeroed page list. The system's fluid page count is adjusted to include the deallocated pages.

If page tables are also reserved for the named memory-resident global section, the reserved memory for the shared page tables is also freed.

If the reserved memory is in use by the named memory-resident global section, the amount of reserved memory not currently in use is freed.

The RESERVED_MEMORY FREE command does not affect the Reserved Memory Registry data file contents, it only affects the memory within the running system.

3.11.2.3. Displaying Reserved Memory

Two different places hold reserved memory information, the Reserved Memory Registry data file and the Reserved Memory Registry in the running system created during system initialization based on the entries in the data file.

Different display mechanisms may be used depending on where the information about the reserved memory originates.

There are three mechanisms for displaying the Reserved Memory Registry within the running system: SYSMAN, the DCL SHOW MEMORY command and SDA.

- SYSMAN

  You can display the Reserved Memory Registry within the running system by issuing the following SYSMAN command:

  ```
  SYSMAN> RESERVED_MEMORY SHOW gs_name /GROUP = n
  ```

  The specified gs_name is the name of the memory-resident global section associated with the entry being displayed from within the running system. If gs_name is not specified, the reserved memory for all registered global sections is displayed.

  The value specified by the /GROUP qualifier is the UIC The value specified by the /GROUP qualifier is the UIC n group number (in octal) associated with the memory-resident section being displayed. If the memory-resident global section is a group global section, you must specify the / GROUP qualifier. If the memory-resident global section is a system global section, you must not specify the /GROUP qualifier. The /GROUP qualifier is allowed only if gs_name is specified.

- DCL SHOW MEMORY command

  You can display the Reserved Memory Registry within the running system by issuing the DCL SHOW MEMORY command. This command shows all memory-related information about the running system including information about the Reserved Memory Registry.

  SHOW MEMORY /RESERVED displays just information about the Reserved Memory Registry within the running system.

  The information displayed by SHOW MEMORY includes how much memory is currently in use by the named global section. It also includes how much memory is reserved and in use (if any) for page tables.

- SDA

  SDA also includes various enhancements to display the Reserved Memory Registry within the running system as well as in crash dump files. The command interface is TBD.

3.11.2.4. Using Reserved Memory

The system services SYS$CREATE_GDZRO and SYS$CRMPSC_GDZRO_64 call internal kernel mode OpenVMS Alpha routines to use the reserved memory registered in the Reserved Memory Registry.

The global section need not be registered in the Reserved Memory Registry. If the global section name is registered in the Reserved Memory Registry, the size of the global section need not exactly match the size of the reserved memory. If the global section is not registered, or if /NOALLOCATE was specified when the global section was registered in the Reserved Memory Registry, the fault option is used for the memory-resident global DZRO section. If the size is greater than the size of the reserved memory, the system service call to create the memory-resident global DZRO section fails if there are not enough additional fluid pages within the system.

If /ALLOCATE was specified when the global section was registered in the Reserved Memory Registry, the allocate option is used for the memory-resident global DZRO section. The size of the global section must be less than or equal to the size of the reserved, preallocated memory or the error SS $_MRES_PFNSMALL is returned by the system service call.

3.11.2.5. Returning Reserved Memory

When a memory-resident global section is deleted, the physical pages used for that global section are deallocated to the free page list if physical pages were not preallocated for this global section. The system's fluid page count is adjusted only for those pages not reserved in the Reserved Memory Registry for this global section.

When a memory-resident global section is deleted, the physical pages used for that global section are returned to the Reserved Memory Registry if physical pages were preallocated for this global section. The physical pages are not deallocated to the free page list and are still reserved. No adjustment is made to the system's fluid page count.

Reserved memory may only be freed to the running system by using the RESERVED_MEMORY FREE command to the SYSMAN utility.

Note

Permanent global sections are deleted by calling SYS$DGBLSC and upon the last reference to the global section. Nonpermanent global sections are simply deleted upon last reference to the global section.

3.11.3. Application Configuration

The configuration of an OpenVMS Alpha application that uses memory-resident global sections performs the following steps:

1. Execute the SYSMAN RESERVED_MEMORY ADD commands that specify the required use of reserved memory.
2. Run AUTOGEN with feedback to set the system's fluid page count appropriately and size the system's pagefile, number of processes and working set maximum size appropriately.
3. Reboot the system to allow for the reserved memory to be deducted from the system's fluid page count and for contiguous, aligned pages to be allocated and zeroed as necessary.

Chapter 4. Managing File System Data Caches

This chapter describes how to manage the Extended File Cache (XFC) (Alpha only) and the Virtual I/O Cache (VIOC). These are the caches that the Files-11 file system uses to cache data for ODS-2 and ODS-5 volumes.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Disabling caching clusterwide	*Section 4.3, "Disabling Caching Clusterwide"*
Mounting a volume with caching disabled	*Section 4.4, "Mounting a Volume With Caching Disabled"*
Managing the Extended File Cache (Alpha Only)	*Section 4.5, "Managing XFC (Alpha Only)"*
Managing the Virtual I/O Cache	*Section 4.6, "Managing the Virtual I/O Cache"*

This chapter explains the following concepts:

Concept	Section
Caching	*Section 4.1, "Understanding Caching"*
File system data caches	*Section 4.2, "Understanding File System Data Caches"*

4.1. Understanding Caching

The Files-11 file system uses a technique called **caching** to improve performance. It keeps a copy of data that it has recently read from disk or written to disk in an area of memory called a **cache**.

When an application reads data, the file system checks whether the data is in its cache. It issues an I/O to read the data from disk only if the data is not in the cache.

Caching improves read performance. Reading data from memory (from the cache) is much faster than reading it from disk.

There are several levels of caching in both the hardware I/O subsystem and in OpenVMS. In general, more levels of caching result in better response time in accessing data.

4.2. Understanding File System Data Caches

For ODS-2 and ODS-5 volumes, the Files-11 file system has several caches. It has **metadata caches** for file metadata such as file headers, and a **data cache** for file data. It can use one of two system-wide data caches, as follows:

File System Data Cache	Description
Virtual I/O Cache (VIOC)	This is the original data cache, and is available on VAX and Alpha.
Extended File Cache (XFC)	This data cache, which is available on OpenVMS Alpha Version 7.3 and later, is available only on OpenVMS Alpha. It provides better performance and more capability than VIOC. XFC is the default cache on OpenVMS Alpha Version 7.3 and later.

This chapter describes how to manage the data caches. For information on managing the metadata caches, see the *OpenVMS Performance Management Manual*. Note that RMS makes use of local and global buffers that can perform data caching. By default, this caching is not enabled. You can manipulate the RMS local and global buffers to affect I/O performance. For information on managing the RMS local and global buffers, see the *Guide to OpenVMS File Applications* [https://docs.vmssoftware.com/guide-to-openvms-file-applications/]. Note also that the modified page list is a type of cache; that is, if a process references a page that is on the modified page list, the page is placed back into the working set of the process and is not output to disk.

Both XFC and VIOC are virtual block caches that maintain consistent data within an OpenVMS Cluster. They cache both data files and image files. The data caches are write-through caches; when an application writes data to a file, the data is written straight through to disk. The application has to wait until the disk I/O completes and the data is on disk.

In an OpenVMS Cluster, different nodes can use different data caches. This allows mixed architecture clusters to benefit from XFC. OpenVMS Alpha nodes can use either XFC or VIOC. OpenVMS VAX nodes can use only VIOC, as described in *Section 4.5.7, "Using XFC in a Mixed Architecture OpenVMS Cluster"*.

XFC improves I/O performance and contains the following features that are not available with VIOC:

- Read-ahead caching
- Automatic resizing of the cache
- Larger maximum cache size
- No limit on the number of closed files that can be cached
- Control over the maximum size of I/O that can be cached
- Control over whether cache memory is static or dynamic

4.3. Disabling Caching Clusterwide

At system startup, the value of a static system parameter controls whether the file system uses a data cache, and if so, which data cache it uses (XFC or VIOC). The system parameter depends on the operating system, as shown in the following table:

Operating System	System Parameter	Enabled	Disabled
OpenVMS Alpha	VCC_FLAGS	1 or 2 (default)[DAG]	0
OpenVMS VAX	VBN_CACHE_S	1 (default)	0

[DAG] 1 selects VIOC, 2 (the default) selects XFC

In an OpenVMS Cluster, if file system data caching is disabled on one node, it is disabled throughout the cluster. The other nodes in the cluster cannot use XFC or VIOC until that node leaves the cluster or reboots with VCC_FLAGS or VBN_CACHE_S set to a non-zero value. You can use the DCL command SHOW MEMORY to determine whether caching is enabled.

To disable caching clusterwide, follow these steps on any node in your OpenVMS Cluster:

1. Set the appropriate system parameter (VCC_FLAGS or VBN_CACHE_S) to 0,using MODPARAMS.DAT.

2. Run AUTOGEN to ensure that other system parameters allow for the new value. This is not essential, but it is advisable.

3. Reboot the system to make the new value effective.

4.4. Mounting a Volume With Caching Disabled

To prevent the file system from caching data on a particular volume, such as a database volume, use the MOUNT /NOCACHE command to mount the volume with caching disabled.

If you are using XFC in an OpenVMS Cluster, mounting a volume /NOCACHE is easier than using SET FILE /CACHING_ATTRIBUTE to set the caching attribute of all the files in the volume to no caching (see *Section 4.5.4, "Disabling Caching for a File"*). Using MOUNT /NOCACHE ensures the minimum caching overhead. Note that the MOUNT /NOCACHE command also disables XQP caching.

Example

This example mounts a database volume labeled ORACLE_VOL1 with caching disabled:

```
$ MOUNT DUA100: ORACLE_VOL1 /NOCACHE /SYSTEM
```

4.5. Managing XFC (Alpha Only)

This section describes how to manage XFC, which is available only on OpenVMS Alpha. It describes the following tasks.

Task	Section
Controlling the size of the cache	*Section 4.5.2, "Controlling the Size of the Cache"*
Controlling the maximum cached I/O size	*Section 4.5.3, "Controlling the*

Task	Section
	Maximum Cached I/O Size"
Disabling caching for a file	*Section 4.5.4, "Disabling Caching for a File"*
Disabling read-ahead caching	*Section 4.5.5, "Disabling Read-Ahead Caching"*
Monitoring performance	*Section 4.5.6, "Monitoring Performance"*
Using in a mixed architecture OpenVMS Cluster	*Section 4.5.7, "Using XFC in a Mixed Architecture OpenVMS Cluster"*

4.5.1. Ensuring that XFC Interoperates with Older Versions

If you have an OpenVMS Cluster system that contains earlier versions of OpenVMS Alpha or OpenVMS VAX and you want to use XFC with OpenVMS Version 7.3 or higher, you must install remedial kits on the systems that are running the earlier versions of OpenVMS. See *VSI OpenVMS Alpha Version 7.3-2 Release Notes* for information on the required kits.

Caution

The remedial kits referred to in the previous section correct errors in the cache-locking protocol of VIOC, the predecessor to XFC, and allow older versions of the caches to operate safely with the new XFC. Without the functionality in the remedial kits, however, the system or processes might hang.

4.5.2. Controlling the Size of the Cache

This section describes how to control the minimum and maximum size of XFC.

XFC is held in virtual memory in S2 space and automatically shrinks and grows, depending on your I/O workload and how much spare memory is available on your system. S2 space is a 64-bit address space, so the cache can grow to very large sizes when required.

As your I/O workload increases, the cache automatically grows, but never to more than the maximum size. And when your applications need memory, the cache automatically shrinks, but never to less than the minimum size.

4.5.2.1. Controlling the Minimum Cache Size

The minimum size of XFC is controlled by the value of the VCC$MIN_CACHE_SIZE entry in the reserved memory registry. VCC$MIN_CACHE_SIZE specifies the amount of memory in megabytes (MB) that is allocated to XFC at system startup. The cache never shrinks below this size. This memory is never released.

Checking the Minimum Size

To check the minimum size of XFC, use either the Sysman utility command RESERVED_MEMORY / SHOW or the DCL command SHOW MEMORY /RESERVED. For example:

```
$ SHOW MEMORY /RESERVED
               System Memory Resources on 11-MAY-2016 15:50:25.64

Memory Reservations (pages):    Group    Reserved      In Use       Type
  VCC$MIN_CACHE_SIZE              ---        1536        1536  Allocated
Total (400.00 Mb reserved)                   1536        1536
```

Setting a Minimum Size

By default, the reserved memory registry does not contain an entry for VCC$MIN_CACHE_SIZE, so no memory is allocated to XFC at system startup. However, XFC allocates a small amount of memory to maintain overall system throughput. The amount of memory allocated varies according to the size of your machine.

To set a minimum size, follow these steps:

1. Use the Sysman utility's RESERVED_MEMORY ADD command to add an entry for VCC $MIN_CACHE_SIZE. For example, to set the minimum size to 300 MB:

   ```
   $ RUN SYS$SYSTEM:SYSMAN
   SYSMAN> RESERVED_MEMORY ADD VCC$MIN_CACHE_SIZE /SIZE=300 /ALLOCATE -
   _SYSMAN> /NOGLOBAL_SECTION /NOZERO /NOPAGE_TABLE
   ```

   You must use all the qualifiers shown in this example. For best performance, set the minimum size to a multiple of 4 MB.

   Note that if you are doing this on a NUMA type machine, where you want to allocate reserved memory in different RADs, you may choose to use the following commands (rather than those above) to set the minimum size:

   ```
   $ RUN SYS$SYSTEM:SYSMAN
   SYSMAN> RESERVED_MEMORY ADD VCC$MIN_CACHE_SIZE /SIZE=300 /ALLOCATE -
   _SYSMAN> /NOGLOBAL_SECTION /NOZERO /NOPAGE_TABLE/RAD=0
   SYSMAN> RESERVED_MEMORY EXTEND VCC$MIN_CACHE_SIZE /SIZE=500 /ALLOCATE -
   _SYSMAN> /NOGLOBAL_SECTION /NOZERO /NOPAGE_TABLE/RAD=1
   ```

2. Run AUTOGEN to ensure that other system parameters allow for the new value. This is not essential, but it is advisable.

3. Reboot the system to make the new value effective.

   During startup, if the system does not have enough memory to allocate the specified minimum size, no memory is allocated to XFC and its minimum size is set to 0 MB.

Changing the Minimum Size

To change the minimum size of XFC, follow these steps:

1. Use the Sysman utility's RESERVED_MEMORY MODIFY command to modify the existing entry for VCC$MIN_CACHE_SIZE. For example, to change the minimum size to 360 MB:

   ```
   $ RUN SYS$SYSTEM:SYSMAN
   ```

```
SYSMAN> RESERVED_MEMORY MODIFY VCC$MIN_CACHE_SIZE /SIZE=360 /ALLOCATE -
_SYSMAN> /NOGLOBAL_SECTION /NOZERO
```

You must use all the qualifiers shown in this example. For best performance, remember to set the minimum size to a multiple of 4 MB.

2. Run AUTOGEN to ensure that other system parameters allow for the new value. This is not essential, but it is advisable.

3. Reboot the system to make the new value effective.

   During startup, if the system does not have enough memory to allocate the specified minimum size, no memory is allocated to XFC and its minimum size is set to 0 MB.

4.5.2.2. Controlling the Maximum Cache Size

To control the maximum size of XFC, use the dynamic system parameter VCC_MAX_CACHE. It specifies the size in megabytes.

By default, VCC_MAX_CACHE is –1, which means that at system startup, the maximum size of XFC is set to 50 percent of the physical memory on the system. For example, if the system has 2 GB of physical memory, its maximum size is set to 1 GB.

Note that the maximum size specified by VCC_MAX_CACHE does not include the memory that XFC consumes indirectly via the OpenVMS lock manager.

The value of VCC_MAX_CACHE at system startup sets an upper limit for the maximum size of XFC. You cannot increase the maximum size beyond that value.

For example, VCC_MAX_CACHE is 60 at system startup, so the maximum size is initially set to 60 MB. You then set VCC_MAX_CACHE to 40, which decreases the maximum size to 40 MB. If XFC is bigger than 40 MB, it gradually shrinks to 40 MB. You then set VCC_MAX_CACHE to 80, but the maximum size is only increased to 60 MB, the value set at system startup. You cannot increase the maximum size beyond the value set at system startup.

If VCC_MAX_CACHE is less than the minimum size specified by the value of the VCC $MIN_CACHE_SIZE entry in the reserved memory registry, then during system startup, VCC_MAX_CACHE is ignored and the maximum size of XFC is set to the same value as the minimum size. In this case, XFC has a fixed size and cannot shrink or grow.

Example

This example reduces the maximum size of XFC on the active system from 60 MB to 40MB:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> USE ACTIVE
SYSGEN> SET VCC_MAX_CACHE 40
SYSGEN> WRITE ACTIVE
$ SET CACHE /RESET
```

The following example makes the change persistent across reboots by changing the current parameter set:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> USE CURRENT
SYSGEN> SET VCC_MAX_CACHE 40
SYSGEN> WRITE CURRENT
```

To make this change permanent, you must enter it into MODPARAMS.DAT. The change will take effect after the next reboot.

4.5.2.3. Enabling a Static Cache Size

On larger machines, XFC may be somewhat limited by the default size of the modified page list. In general, the modified page list can be considered a 1 to 1 correspondence data cache. XFC is a many to one cache; that is, in general, a cached page is accessed by many users. On large memory systems, AUTOGEN usually sets MPW_HILIMIT to a very large value. This may mean that there are not enough free pages for the memory management subsystem to give to XFC.

You can force XFC to have a minimum number of pages as specified in *the section called "Setting a Minimum Size"*. You can also permanently allocate memory just for XFC, which prevents any overhead caused by the dynamic allocation and deallocation routines (similar in operation to VIOC). To do this, set system parameter VCC_MAX_CACHE to be equal to the memory reservation specified by VCC $MIN_CACHE_SIZE.

For example, if your system has 128 GB of memory, you may find that by dedicating 8 GB of memory to XFC, your cache hit rates and overall response time is consistently good. For example:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> USE CURRENT
SYSGEN> SET VCC_MAX_CACHE 8000
SYSGEN> WRITE ACTIVE
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> RESERVED_MEMORY ADD VCC$MIN_CACHE_SIZE /SIZE=8000 /ALLOCATE -
_SYSMAN> /NOGLOBAL_SECTION /NOZERO /NOPAGE_TABLE/RAD=0
```

To make the change to VCC_MAX_CACHE permanent, you must enter it into MODPARAMS.DAT. This change will take effect after the next reboot. VSI recommends that you run AUTOGEN to ensure that other system parameters allow for the new value.

4.5.3. Controlling the Maximum Cached I/O Size

The dynamic system parameter VCC_MAX_IO_SIZE controls the maximum size of I/O that can be cached by XFC. This parameter specifies the size in blocks. By default, it is 128.

Example

This example changes the maximum size of I/O that can be cached by XFC to 1,000 blocks:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> USE ACTIVE
SYSGEN> SET VCC_MAX_IO_SIZE 1000
SYSGEN> WRITE ACTIVE
```

This series of commands affects I/Os to volumes currently mounted on the local node, as well as to volumes mounted in the future. After you enter these commands, XFC does not cache I/Os that are larger than 1,000 blocks. The SHOW MEMORY /CACHE /FULL command provides a histogram of I/O sizes that can provide guidelines for efficient parameter setting.

4.5.4. Disabling Caching for a File

To prevent XFC to from caching a particular file, such as a database file, set the caching attribute of the file to no caching.

The **caching attribute** of a file is the default caching option that is used by XFC when an application accesses the file without specifying which caching option it wants to use. The caching option can be either write-through caching or no caching.

If you want a file to be cached, set its caching attribute to write-through (the default). If you do not want a file to be cached, set its caching attribute to no caching.

Use...	To...
SET FILE /CACHING_ATTRIBUTE=keyword [1]	Set the caching attribute of a file or directory
DIRECTORY /CACHING_ATTRIBUTE or DIRECTORY /FULL	Show the caching attribute of a file or directory
DIRECTORY /SELECT=CACHING_ATTRIBUTE=(keyword[,...])[1]	Show all the files and directories that have a particular caching attribute

[1]keyword can be either WRITETHROUGH or NO_CACHING.

XFC does not cache directories. The caching attribute of a directory controls only how the caching attribute is inherited by new files and subdirectories created in the directory:

- When you create a new directory or file, it inherits its caching attribute from its parent directory.
- When you create a new version of an existing file, the new file inherits its caching attribute from the highest version of the existing file.

Examples

- This example sets the caching attribute to no caching for all the files in and under the directory [SMITH.BORING].

  The first SET FILE command sets the attribute for the directory to make sure that all files and subdirectories subsequently created in it inherit the attribute. The second SET FILE command sets the attribute for all existing files and directories in and under the directory.

  ```
  $ SET FILE DISK$USERS:[SMITH]BORING.DIR;1 /CACHING_ATTRIBUTE=NO_CACHING
  $ SET FILE DISK$USERS:[SMITH.BORING...]*.*;* /
  CACHING_ATTRIBUTE=NO_CACHING
  ```

- This example uses the DIRECTORY command's /CACHING_ATTRIBUTE qualifier to show the caching attribute of MYFILE.TXT:

  ```
  $ DIRECTORY MYFILE.TXT /CACHING_ATTRIBUTE
  Directory DISK$USERS:[SMITH]

  MYFILE.TXT;1          Write-through

  Total of 1 file.
  ```

- This example shows all the files in the volume DISK$USERS that have a caching attribute of no caching.

  ```
  $ DIRECTORY DISK$USERS:[000000...]*.* /
  SELECT=CACHING_ATTRIBUTE=NO_CACHING
  ```

4.5.5. Disabling Read-Ahead Caching

XFC uses a technique called **read-ahead caching** to improve the performance of applications that read data sequentially. It detects when a file is being read sequentially in equal-sized I/Os, and fetches data ahead of the current read, so that the next read instruction can be satisfied from cache.

To disable read-ahead caching on the local node, set the dynamic system parameter VCC_READAHEAD to 0. By default, this parameter is 1, which allows the local node to use read-ahead caching.

Example

This example disables read-ahead caching on the local node:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> USE CURRENT
SYSGEN> SET VCC_READAHEAD 0
SYSGEN> WRITE ACTIVE
```

This series of commands affects volumes currently mounted on the local node, as well as volumes mounted in the future. Once you enter these commands, read-ahead caching is not used on the local node.

4.5.6. Monitoring Performance

XFC provides more information than VIOC. For example, you can obtain information on system-wide, volume-wide, or even a per-file basis. Disk I/O response times are also available. See the *VSI OpenVMS DCL Dictionary* for a description of the SHOW MEMORY command.

4.5.6.1. System-Wide Statistics

Use SHOW MEMORY /CACHE to monitor the overall system performance of XFC. For example:

```
$ SHOW MEMORY /CACHE

              System Memory Resources on 26-JAN-2017 15:58:18.71

Extended File Cache  (Time of last reset: 24-JAN-2017 15:03:39.05)
 Allocated (Mbytes)      ❶  3000.00  Maximum size (Mbytes)    ❷  5120.00
 Free (Mbytes)           ❸  2912.30  Minimum size (Mbytes)    ❹  3000.00
 In use (Mbytes)         ❺    87.69  Percentage Read I/Os     ❻      98%
 Read hit rate           ❼      92%  Write hit rate           ❽       0%
 Read I/O count          ❾  178136  Write I/O count          ❿    1867
 Read hit count          ⓫  165470  Write hit count          ⓬       0
 Reads bypassing cache   ⓭    2802  Writes bypassing cache  ⓮      39
 Files cached open       ⓯     392  Files cached closed     ⓰     384
 Vols in Full XFC mode   ⓱       0  Vols in VIOC Compatible mode  ⓲  4
 Vols in No Caching mode ⓳       1  Vols in Perm. No Caching mode ⓴  0
```

❶ Allocated	The amount of memory currently allocated to the cache.
❸ Free	The amount of memory currently allocated to the cache that is not being used.
❺ In Use	The amount of memory currently allocated to the cache that is being used. This is the difference between the Allocated value and the Free value.

❼ Read hit rate	The ratio of the Read hit count field divided by the Read I/O count field.
❾ Read I/O count	The total number of read I/Os seen by the cache since system startup.
❶❶ Read hit count	The total number of read hits since system startup. A read hit is a read I/O that did not require a physical I/O to disk because the data was found in the cache.
❶❸ Reads bypassing cache	The total number of read I/Os since system startup that were seen by the cache but were not cached, for example, because they were too big, or they were for volumes mounted /NOCACHE, or they specified one of the following QIO modifiers: IO$M_DATACHECK, IO$M_INHRETRY, or IO$M_NOVCACHE.
❶❺ Files cached open	The number of open files currently being cached.
❶❼ Volumes in Full XFC mode	Reserved for VSI. Should be 0.
❶❾ Volumes in No Caching mode	If caching is disabled on the local node or on another node in the OpenVMS Cluster, this is the number of volumes that are currently mounted on the local node. Otherwise, it is zero.
❷ Maximum size	The maximum size that the cache could ever grow to.
❹ Minimum size	The minimum size that the cache could ever shrink to. This is controlled by the value of the VCC$MIN_CACHE_SIZE entry in the reserved memory registry.
❻ Percentage Read I/Os	Percentage of I/Os that are reads.
❽ Write hit rate	This field is reserved for future use.
❿ Write I/O count	The total number of write I/Os seen by the cache since system startup.
❶❷ Write hit count	This field is reserved for future use.
❶❹ Writes bypassing cache	The total number of write I/Os since system startup that were seen by the cache but were not cached, for example, because they were too big, or they were for volumes mounted /NOCACHE, or they specified one of the following QIO modifiers: IO$M_DATACHECK, IO$M_ERASE, IO $M_INHRETRY, or IO$M_NOVCACHE.
❶❻ Files cached closed	The number of closed files that still have valid data in the cache.
❶❽ Volumes in VIOC compatible mode	The number of volumes being cached by XFC that are using the VCC caching protocol. As of OpenVMS Version 7.3, XFC uses only VCC caching protocol.
❷❿ Vols in Perm. No Caching mode	This field should be zero. If nonzero, XFC has detected an illegal write operation to this device and has disabled caching to this device.

See the *VSI OpenVMS DCL Dictionary* for a description of the SHOW MEMORY command.

4.5.7. Using XFC in a Mixed Architecture OpenVMS Cluster

In an OpenVMS Cluster, some nodes can use XFC and other nodes can use VIOC. This allows mixed architecture clusters to benefit from XFC.

When a volume is mounted on a node that is using VIOC, the nodes using XFC cannot cache any files in the volume that are shared for writing. A file that is **shared for writing** is one that is being accessed by more than one node in an OpenVMS Cluster, and at least one of those nodes opened it for write access.

4.6. Managing the Virtual I/O Cache

This section describes how to manage VIOC. It describes the following tasks:

Task	Section
Selecting VIOC on an Alpha system	*Section 4.6.2, "Selecting VIOC on an Alpha System"*
Controlling the size of the cache	*Section 4.6.3, "Controlling the Size of the Cache"*
Monitoring performance	*Section 4.6.4, "Displaying VIOC Statistics"*

The virtual I/O cache is a clusterwide, write-through, file-oriented, disk cache that can reduce the number of disk I/O operations and increase performance. The purpose of the virtual I/O cache is to increase system throughput by reducing file I/O response times with minimum overhead. The virtual I/O cache operates transparently of system management and application software, and maintains system reliability while it significantly improves virtual disk I/O read performance.

4.6.1. Understanding How the Cache Works

The virtual I/O cache can store data files and image files. For example, ODS-2 disk file data blocks are copied to the virtual I/O cache the first time they are accessed. Any subsequent read requests of the same data blocks are satisfied from the virtual I/O cache (hits) eliminating any physical disk I/O operations (misses) that would have occurred.

Depending on your system work load, you should see increased application throughput, increased interactive responsiveness, and reduced I/O load.

Note

Applications that initiate single read and write requests do not benefit from virtual I/O caching as the data is never reread from the cache. Applications that rely on implicit I/O delays might abort or yield unpredictable results.

Several policies govern how the cache manipulates data, as follows:

- Write-through—All write I/O requests are written to the cache as well as to the disk.
- Least Recently Used (LRU)—If the cache is full, the least recently used data in the cache is replaced.
- Cached data maintained across file close—Data remains in the cache after a file is closed.
- Allocate on read and write requests—Cache blocks are allocated for read and write requests.

4.6.2. Selecting VIOC on an Alpha System

If for some reason, you want an Alpha system to use VIOC instead of XFC, follow these steps:

1. Remove the entry for VCC$MIN_CACHE_SIZE from the reserved memory registry, using the Sysman utility's RESERVED_MEMORY REMOVE command:

   ```
   $ RUN SYS$SYSTEM:SYSMAN
   SYSMAN> RESERVED_MEMORY REMOVE VCC$MIN_CACHE_SIZE /NOGLOBAL_SECTION
   ```

   This makes sure that no memory is allocated to XFC in Step 4, when the system reboots with VIOC.

2. Set the VCC_FLAGS system parameter to 1.

3. Run AUTOGEN to ensure that other system parameters allow for the new value. This is not essential, but it is advisable.

4. Reboot the system. VIOC is automatically loaded during startup instead of XFC, because VCC_FLAGS is 1.

If you forgot to remove the VCC$MIN_CACHE_SIZE entry from the reserved memory registry in Step 1, memory is allocated to XFC even though XFC is not loaded. Nothing can use this memory. If this happens, use the Sysman utility's RESERVED_MEMORY FREE command to release this memory:

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> RESERVED_MEMORY FREE VCC$MIN_CACHE_SIZE /NOGLOBAL_SECTION
```

4.6.3. Controlling the Size of the Cache

The way that you control the size of VIOC depends on whether you have an OpenVMS Alpha or OpenVMS VAX system.

OpenVMS Alpha

On OpenVMS Alpha systems, the size of VIOC is fixed at system startup time. The cache can not shrink or grow. The value of the static system parameter VCC_MAXSIZE specifies the size of the cache in blocks. By default it is 6400 blocks (3.2 MB).

To change the size of VIOC on an OpenVMS Alpha system, follow these steps:

1. Set the VCC_MAXSIZE system parameter to the required value.

2. Run AUTOGEN to ensure that other system parameters allow for the new value. This is not essential, but it is advisable.

3. Reboot the system to make the new value effective.

OpenVMS VAX

On OpenVMS VAX systems, you can use the static system parameter VCC_PTES to specify the maximum size of VIOC. This parameter specifies the size in pages. By default it is 2,000,000,000.

VIOC automatically shrinks and grows, depending on your I/O workload and how much spare memory is available on your system. As your I/O workload increases, the cache automatically grows, but never to more than the maximum size. And when your applications need memory, the cache automatically shrinks.

To change the maximum size of VIOC on an OpenVMS VAX system, follow these steps:

1. Set the VCC_MAXSIZE system parameter to the required value.

2. Run AUTOGEN to ensure that other system parameters allow for the new value. This is not essential, but it is advisable.

3. Reboot the system to make the new value effective.

4.6.4. Displaying VIOC Statistics

Use the DCL command SHOW MEMORY/CACHE/FULL to display statistics about the virtual I/O cache, as shown in the following example:

```
$ SHOW MEMORY/CACHE/FULL
            System Memory Resources on 10-OCT-2017 18:36:12.79
Virtual I/O Cache
Total Size (pages)    ❶   2422    Read IO Count           ❷     9577
Free Pages            ❸   18      Read Hit Count          ❹     5651
Pages in Use          ❺   2404    Read Hit Rate           ❻     59%
Maximum Size (SPTEs)  ❼   11432   Write IO Count          ❽     2743
Files Retained        ❾   99      IO Bypassing the Cache  ❿     88
```

Note

This example shows the output for the SHOW MEMORY/CACHE/FULL command on a VAX system. The SHOW MEMORY/CACHE/FULL command displays slightly different fields on an Alpha system.

❶ Total Size	Displays the total number of system memory pages that VIOC currently controls.
❸ Free Pages	Displays the number of pages controlled by VIOC that do not contain cache data.
❺ Pages in Use	Displays the number of pages controlled by VIOC that contain valid cached data.
❼ Maximum Size	Shows the maximum size that the cache could ever grow to.
❾ Files Retained	Displays the number of files that are closed but the file system control information is being retained because they have valid data residing in the cache.
❷ Read I/O Count	Displays the total number of read I/Os that have been seen by VIOC since the last system.
❹ Read Hit Count	Displays the total number of read I/Os that did not do a physical I/O because the data for them was found in the cache since the last system BOOT.
❻ Read Hit Rate	Displays the read hit count and read I/O count ratio.
❽ Write I/O Count	Shows the total number of write I/Os that have been seen by the cache since the last system BOOT.
❿ I/O Bypassing	Displays the count of I/Os that for some reason did not attempt to satisfy the request/update by the cache.

4.6.5. Enabling VIOC

By default, virtual I/O caching is enabled. Use the following system parameters to enable or disable caching. Change the value of the parameters in MODPARAMS.DAT, as follows:

Parameter	Enabled	Disabled
VCC_FLAGS (Alpha)	1	0
VBN_CACHE_S (VAX)	1	0

Once you have updated MODPARAMS.DAT to change the value of the appropriate parameter, you must run AUTOGEN and reboot the node or nodes on which you have enabled or disabled caching. Caching is automatically enabled or disabled during system initialization. No further user action is required.

4.6.6. Determining If VIOC Is Enabled

SHOW MEMORY/CACHE indicates whether VIOC caching is on or off on a running system. (This is a lot easier than using SYSGEN.)

SYSGEN can be used to examine parameters before a system is booted. For example, you can check the system parameter VCC_FLAGS (on Alpha) or VBN_CACHE_S (on VAX) to see if virtual I/O caching is enabled by using SYSGEN, as shown in the following Alpha example:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> SHOW VCC_FLAGS
```

A value of 0 indicates that caching is disabled; the value 1 indicates caching is enabled.

4.6.7. Memory Allocation and VIOC

The memory allocated to caching is determined by the size of the free-page list. The size of the virtual I/O cache can grow if one of the following conditions is true:

- If the amount of available free memory is twice the value of FREEGOAL and if proactive memory reclamation is enabled for periodically waking processes.
- If the amount of available free memory is equal to the value of FREEGOAL and if proactive memory reclamation is enabled for long-waiting processes.
- If the amount of available free memory is greater than GROWLIM and if proactive memory reclamation is not enabled.

The cache size is also limited by the following:

- The number of system page table entries (SPTE) that are available. This number is a calculated value determined at boot time.
- The demands of the memory management subsystem. The memory management subsystem has a direct interface to cache so that, when necessary, it can demand that the cache return space to it.

How is memory reclaimed from the cache? The swapper can reclaim memory allocated to the virtual I/O cache by using first-level trimming. In addition, a heuristic primitive shrinks the cache returning memory in small increments.

4.6.8. Adjusting VIOC Size

The size of the virtual I/O cache is controlled by the system parameter VCC_MAXSIZE. The amount of memory specified by this parameter is statically allocated at system initialization and remains owned by the virtual I/O cache.

To increase or decrease the size of the cache, modify VCC_MAXSIZE and reboot the system.

4.6.9. VIOC and OpenVMS Cluster Configurations

The cache works on all supported configurations from single-node systems to large mixed-interconnect OpenVMS Cluster systems. The virtual I/O cache is nodal; that is, the cache is local to each OpenVMS Cluster member. Any base system can support virtual I/O caching; an OpenVMS Cluster license is not required to use the caching feature.

Note

If any member of an OpenVMS Cluster does not have caching enabled, then no caching can occur on any node in the OpenVMS Cluster (including the nodes that have caching enabled). This condition remains in effect until the node or nodes that have caching disabled either enable caching or leave the cluster.

The lock manager controls cache coherency. The cache is flushed when a node leaves the OpenVMS Cluster. Files opened on two or more nodes with write access on one or more nodes are not cached.

Chapter 5. Testing the System with UETP

This chapter explains how to use UETP (user environment test package) to test whether the OpenVMS operating system is installed correctly.

5.1. Overview

This overview summarizes what UETP does and how you use it. The rest of the chapter provides detailed instructions for setting up your system for testing, running the tests, and troubleshooting errors.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Running UETP (a summary)	*Section 5.1.2, "Summary of How to Use UETP"*
Preparing to use UETP	*Section 5.2, "Preparing to Use UETP"*
Setting up the devices to be tested	*Section 5.3, "Setting Up the Devices to Be Tested"*
Starting UETP	*Section 5.4, "Starting UETP"*
Stopping a UETP operation	*Section 5.5, "Stopping a UETP Operation"*
Troubleshooting: identifying and solving problems	*Section 5.7, "Troubleshooting: Possible UETP Errors"*

This chapter explains the following concepts:

Concept	Section
Understanding UETP	*Section 5.1.1, "Understanding UETP"*
Troubleshooting (an overview)	*Section 5.6, "Troubleshooting: An Overview"*

Concept	Section
UETP Tests and Phases	*Section 5.8, "UETP Tests and Phases"*

5.1.1. Understanding UETP

UETP is a software package designed to test whether the OpenVMS operating system is installed correctly. UETP puts the system through a series of tests that simulate a typical user environment by making demands on the system that are similar to demands that can occur in everyday use.

UETP is not a diagnostic program; it does not attempt to test every feature exhaustively. When UETP runs to completion without encountering nonrecoverable errors, the system being tested is ready for use.

UETP exercises devices and functions that are common to all OpenVMS systems, with the exception of optional features such as high-level language compilers. The system components tested include the following ones:

- Most standard peripheral devices
- System's multiuser capability
- DECnet for OpenVMS software
- Clusterwide file access and locks

5.1.2. Summary of How to Use UETP

This section summarizes the procedure for running all phases of UETP with default values. If you are familiar with the test package, refer to this section. If you want additional information, refer to *Section 5.2, "Preparing to Use UETP"*.

Note

If you are using UETP on an OpenVMS Alpha or I64 system, you must execute the CREATE_SPECIAL_ACCOUNTS.COM command procedure to create the SYSTEST and SYSTEST_CLIG accounts before you begin the following procedure. For complete information about the CREATE_SPECIAL_ACCOUNTS.COM command procedure, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

---

1. Log in to the SYSTEST account as follows:

   ```
   Username: SYSTEST
   Password:
   ```

   ### Caution

   Because the SYSTEST and SYSTEST_CLIG accounts have privileges, unauthorized use of these accounts can compromise the security of your system.

   ---

2. Make sure no user programs are running and no user volumes are mounted.

Caution

By design, UETP assumes and requests the exclusive use of system resources. If you ignore this restriction, UETP can interfere with applications that depend on these resources.

3. After you log in, check all devices to be sure that the following conditions exist:

   - All devices you want to test are powered up and are on line to the system.
   - Scratch disks are mounted and initialized.
   - Disks contain a directory named [SYSTEST] with OWNER_UIC=[1, 7]. (You can create this directory with the DCL command CREATE /DIRECTORY.)
   - Scratch magnetic tape reels are *physically* mounted on each drive you want tested and are initialized with the label UETP (using the DCL command INITIALIZE). Make sure magnetic tape reels contain at least 600 feet of tape.
   - Scratch tape cartridges have been inserted in each drive you want to test and are initialized with the label UETP (using the DCL command INITIALIZE).
   - Line printers and hardcopy terminals have plenty of paper.
   - Terminal characteristics and baud rate are set correctly. (Refer to the user's guide for your terminal.)

   Note that some communication devices need to be set up by a VSI support representative. (See *Section 5.3, "Setting Up the Devices to Be Tested"*.)

   If you encounter any problems in preparing to run UETP, read *Section 5.3, "Setting Up the Devices to Be Tested"* before proceeding.

4. To start UETP, enter the following command and press Return:

   ```
   $ @UETP
   ```

   UETP responds with the following question:

   ```
   Run "ALL" UETP phases or a "SUBSET" [ALL]?
   ```

   Press Return to choose the default response enclosed in brackets. UETP responds with the following sequence of questions:

   ```
   How many passes of UETP do you wish to run [1]?
   How many simulated user loads do you want [4]?
   Do you want Long or Short report format [Long]?
   ```

   Press Return after each prompt. After you answer the last question, UETP initiates its entire sequence of tests, which run to completion without further input. The final message should look like the following one:

   ```
   ************************************************
   *                                              *
      END OF UETP PASS 1 AT 22-JUN-1998 16:30:09.38
   *                                              *
   ```

```
**************************************************
```

Note

If you want to run UETP without using the default responses, refer to *Section 5.4, "Starting UETP"*, which explains your options.

5. After UETP runs, check the log files for errors. If testing completes successfully, the OpenVMS operating system is in proper working order.

   If UETP does not complete successfully, refer to *Section 5.6, "Troubleshooting: An Overview"* for information about troubleshooting.

Note

After a run of UETP, you should run the Error Log utility to check for hardware problems that can occur during a run of UETP. For information about running the Error Log utility, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

5.2. Preparing to Use UETP

This section contains detailed instructions for running UETP, including:

- Logging in
- Using the [SYSTEST] directory

5.2.1. Logging In

Obtain the SYSTEST password from your system manager. Log in to the SYSTEST account from the console terminal as follows:

```
Username: SYSTEST
Password:
```

Note

Because SYSTEST has privileges, unauthorized use of this account can compromise the security of your system.

UETP will fail if you do not run the test from the SYSTEST account. Also, if you try to run UETP from a terminal other than the console terminal, the device test phase displays an error message stating that the terminal you are using is unavailable for testing. You can ignore this message.

After you log in to the SYSTEST account, enter the command SHOW USERS to make sure no user programs are running and no user volumes are mounted. UETP requires exclusive use of system resources. If you ignore this restriction, UETP can interfere with applications that depend on these resources.

Note

The information contained in *Section 5.7.2, "Wrong Quotas, Privileges, or Account"* can help you identify and solve problems, including wrong quotas, privileges, or accounts, that could occur when you are running UETP. Refer to this section before you run UETP.

5.2.2. Using the SYSTEST Directories

If you logged in successfully, your default directory is [SYSTEST] on the system disk. UETP uses this directory to hold all the files used by UETP command procedure (UETP.COM) and temporary files used by UETP during testing.

On a typical system, the DCL command SHOW LOGICAL displays the translation of the logical name SYS$TEST:

```
$ SHOW LOGICAL SYS$TEST
   "SYS$TEST" = "SYS$SYSROOT:[SYSTEST]"  (LNM$SYSTEM_TABLE)
```

To use UETP to test a particular disk, such as a scratch disk, create either a [SYSTEST] directory or a [SYS0.SYSTEST] directory on that disk. *Section 5.3.3, "How UETP Works on Disks"* discusses setting up scratch disks for testing.

5.3. Setting Up the Devices to Be Tested

After you log in, set up the devices on the system for UETP testing, as described in the following sections. Note that your system might not have all the devices described in this section.

5.3.1. Check Your Devices

Examine all devices that UETP will use to be sure that the following conditions exist:

- All devices you want to test are turned on and are on line.
- Scratch disks are initialized and mounted.
- Disks contain a directory named [SYSTEST] with OWNER_UIC=[1, 7]. Use the CREATE/DIRECTORY command if the [SYSTEST] directory does not exist on the disk.
- Scratch magnetic tape reels are *physically* mounted on each drive you want tested and are initialized with the label UETP (using the DCL command INITIALIZE). Make sure magnetic tape reels contain at least 600 feet of tape.
- Scratch tape cartridges have been inserted in each drive you want to test and are mounted and initialized with the label UETP (using the DCL command INITIALIZE).
- Line printers and hardcopy terminals have plenty of paper.
- Terminal characteristics and baud rate are set correctly. (Refer to the user's guide for your terminal).

Note that some communications devices discussed in this section must be set up by a VSI support representative.

5.3.2. System Disk Space Required

Before running UETP, be sure that the system disk has at least 1200 blocks available. Note that systems running more than 20 load test processes can require a minimum of 2000 available blocks. If you run multiple passes of UETP, log files will accumulate in the default directory and further reduce the amount of disk space available for subsequent passes.

If disk quotas are enabled on the system disk, disable them before you run UETP.

5.3.3. How UETP Works on Disks

The disk test phase of UETP uses most of the available free space on each testable disk in the following manner:

- On each testable disk, the device test phase tries to create two files. The size of these files depends on how much free space is available on the disk. Usually the test creates each file with 0.1% of the free space on the disk. However, if the disk is almost full, the test creates files that are 5 blocks. If the test cannot create 5 block files, it fails. Only the initial file creation can cause the device test to fail because it lacks disk space.
- The test randomly reads and writes blocks of data to the files. After every multiple of 20 writes for each file, the test tries to extend the file. The size of this extension is either 5% of the free disk space or 5 blocks if the file was created with 5 blocks. This process of extension continues until the combined space of the files reaches 75% of the free disk space.

By creating and extending fragmented files in this way, UETP exercises the disk. This allows the test to check for exceeded quotas or a full disk, and to adjust for the amount of available disk space.

As with other disks, shadow sets and volume sets can be tested with UETP;the expectation is that the individual members will be listed as untestable during UETINIDEV (initialization of UETP). UETINIDEV lists errors when testing using a shadow set during the system disk (UETDISK00) pass, however, the shadow set is listed as testable. When testing using a volume set, errors will be noted against all but relative volume number 1, and all but relative volume 1 will be listed as untestable at the end of UETINIDEV.

5.3.4. Prepare Disk Drives

To prepare each disk drive in the system for UETP testing, use the following procedure:

1. Place a scratch disk in the drive and spin up the drive. If a scratch disk is not available, use any disk with a substantial amount of free space; UETP does not overwrite existing files on any volume. If your scratch disk contains files that you want to keep, do not initialize the disk; go to step 3.
2. If the disk does not contain files you want to save, initialize it. For example:

   ```
   $ INITIALIZE DUA1: TEST1
   ```

   This command initializes DUA1 and assigns the volume label TEST1 to the disk. All volumes must have unique labels.
3. Mount the disk. For example:

   ```
   $ MOUNT/SYSTEM DUA1: TEST1
   ```

   This command mounts the volume labeled TEST1 on DUA1. The /SYSTEM qualifier indicates that you are making the volume available to all users on the system.

4. UETP uses the [SYSTEST] directory when testing the disk. If the volume does not contain the directory [SYSTEST], you must create it. For example:

   ```
   $ CREATE/DIRECTORY/OWNER_UIC=[1, 7] DUA1:[SYSTEST]
   ```

   This command creates a [SYSTEST] directory on DUA1 and assigns a user identification code (UIC) of [1, 7]. The directory must have a UIC of [1, 7] to run UETP.

If the disk you have mounted contains a root directory structure, you can create the [SYSTEST] directory in the [SYS0.] tree.

5.3.5. Magnetic Tape Drives

Set up magnetic tape drives that you want to test by performing the following steps:

1. Place a scratch magnetic tape with at least 600 feet of magnetic tape in the tape drive. Make sure that the write-enable ring is in place.
2. Position the magnetic tape at the BOT (beginning-of-tape) and put the drive on line.
3. Initialize each scratch magnetic tape with the label UETP. For example, if you have physically mounted a scratch magnetic tape on MUA1, enter the following command and press Return:

   ```
   $ INITIALIZE MUA1: UETP
   ```

   Magnetic tapes must be labeled UETP to be tested. As a safety feature, UETP does not test tapes that have been mounted with the MOUNT command.

If you encounter a problem initializing the magnetic tape or if the test has a problem accessing the magnetic tape, refer to the description of the INITIALIZE command in the *VSI OpenVMS DCL Dictionary*.

5.3.6. Tape Cartridge Drives

To set up tape cartridge drives you want to test, perform the following steps:

1. Insert a scratch tape cartridge in the tape cartridge drive.
2. Initialize the tape cartridge. For example:

   ```
   $ INITIALIZE MUA0: UETP
   ```

   Tape cartridges must be labeled UETP to be tested. As a safety feature, UETP does not test tape cartridges that have been mounted with the MOUNT command.

If you encounter a problem initializing the tape cartridge, or if the test has a problem accessing the tape cartridge, refer to the description of the DCL INITIALIZE command in the *VSI OpenVMS DCL Dictionary*.

TLZ04 Tape Drives

During the initialization phase, UETP sets a time limit of 6 minutes for a TLZ04 unit to complete the UETTAPE00 test. If the device does not complete the UETTAPE00 test within the allotted time, UETP displays a message similar to the following one:

```
-UETP-E-TEXT, UETTAPE00.EXE testing controller MKA was stopped ($DELPRC)
```

```
        at 16:23:23.07 because the time out period (UETP$INIT_TIMEOUT)
        expired or because it seemed hung or because UETINIT01 was aborted.
```

To increase the timeout value, enter a command similar to the following one before running UETP:

```
$ DEFINE/GROUP UETP$INIT_TIMEOUT "0000 00:08:00.00"
```

This example defines the initialization timeout value to 8 minutes.

5.3.7. Compact Disc Drives

To run UETP on an RRD40 or RRD50 compact disc drive, you must first load the test disc that you received with your compact disc drive unit.

5.3.8. Optical Disk Drives

To run UETP on an RV60 drive, set up the RV64 optical disk-storage system, perform the following steps:

1. Use the Jukebox Control Software (JCS) to load an optical disk in each of the RV60 drives. JCS is a layered product on the OpenVMS operating system that comes with the RV64 and is responsible for controlling the robot arm that loads and unloads the disks.

2. Initialize the optical disks with the label UETP, but do not mount them.

UETP tests all the RV60s present in the RV64 simultaneously. Unlike the tape tests, UETP does not reinitialize the optical disks at the end of the test.

5.3.9. Terminals and Line Printers

Terminals and line printers must be turned on and on line to be tested by UETP. Check that line printers and hardcopy terminals have enough paper. The amount of paper required depends on the number of UETP passes that you plan to execute. Each pass requires two pages for each line printer and hardcopy terminal.

Check that all terminals are set to the correct baud rate and are assigned appropriate characteristics. (Refer to the user's guide for your terminal.)

Spooled devices and devices allocated to queues fail the initialization phase of UETP and are not tested.

5.3.10. Ethernet Adapters

Make sure that no other processes are sharing the Ethernet adapter device when you run UETP.

Note

UETP will not test your Ethernet adapter if DECnet for OpenVMS or some other application has the device allocated.

Because either DECnet for OpenVMS or the LAT terminal server can try to use the Ethernet adapter (a shareable device), you must shut down DECnet and the LAT terminal server before you run the device test phase, if you want to test the Ethernet adapter.

5.3.11. DR11–W Data Interface (VAX Only)

The DR11–W data interface uses an internal logical loopback mode that tests all features except that of module connectors, cables, and transceivers.

Caution

Only a VSI support representative can set up the DR11–W data interface for UETP testing.

Because random external patterns are generated during this operation, the user device or other processor might need to be isolated from the DR11–W data interface being tested until the testing is completed.

To test the DR11–W data interface properly, the E105 switch pack must be set as follows:

Switch 1	Switch 2	Switch 3	Switch 4	Switch 5
Off	On	Off	Off	On

When UETP testing is completed, restore the DR11–W data interface to the proper operating configuration.

5.3.12. DRV11–WA Data Interface (VAX Only)

The DRV11–WA data interface is a general-purpose, 16-bit, parallel, direct memory access (DMA) data interface.

Caution

Only a VSI support representative can set up the DRV11–WA data interface for UETP testing.

To prepare the DRV11–WA driver on a MicroVAX computer for UETP testing, be sure the following conditions exist:

- The jumpers on the DRV11–WA board are set to W2, W3, and W6.
- A loopback cable is connected to the DRV11–WA board.
- The DRV11–WA board occupies slots 8 to 12. If the DRV11–WA is in another location, timeout errors can occur.

When UETP testing is completed, restore the DRV11–WA to the proper operating configuration.

5.3.13. DR750 or DR780 (DR32 Interface) (VAX Only)

The DR32 (DR750 or DR780) device is an interface adapter that connects the internal memory bus of a VAX processor to a user-accessible bus called the DR32 device interconnect (DDI).

Caution

Only a VSI support representative can set up the DR750 or DR780 for UETP testing.

To prepare the DR750 or the DR780 for UETP testing, use the following procedure:

1. Copy the DR780 microcode file, XF780.ULD, from the diagnostic medium to SYS$SYSTEM. Use the procedure described in the documentation provided with the DR780 Microcode Kit.
2. Turn off the power to the DR780.
3. Make the following DR780 backplane jumper changes:
   a. Remove the jumper from W7 and W8.
   b. Add a jumper from E04M1 to E04R1.
   c. Add a jumper from E04M2 to E04R2.
4. Disconnect the DDI cable from the DR780. This cable is either a BC06V–nn cable, which can be disconnected, or a BC06R–nn cable, which requires that you remove its paddle card from the backplane of the DR780.
5. Restore power to the DR780.

When UETP testing is completed, restore the DR750 or the DR780 to the proper operating configuration.

5.3.14. Second LPA11–K Device

If you have two LPA11–K devices, be sure that each is given a systemwide logical name in the SYS $MANAGER:LPA11STRT.COM file. The logical name for the first LPA11–K device should be LPA11$0, and the logical name for the second LPA11–K device should be LPA11$1.

5.3.15. Devices That Are Not Tested

UETP does not test the following devices; their status has no effect on UETP execution:

- Devices that require operator interaction (such as card readers)
- Software devices (such as the null device and local memory mailboxes)

UETP does not have specific tests for UDA, HSC, or CI devices; they are tested implicitly by the disk, magnetic tape, and DECnet for OpenVMS tests.

UETP also does not test the console terminal or console drives. If you boot the system, log in, and start UETP, you have shown that these devices can be used.

5.3.16. OpenVMS Cluster Testing

Before you run UETP in an OpenVMS Cluster environment, check the SYSTEST_CLIG account. The SYSTEST_CLIG account parallels SYSTEST except that it is dedicated to running the cluster-integration test. The requirements for the SYSTEST_CLIG account are as follows:

- The account should be present in the user authorization file, exactly as distributed by VSI on each system in your OpenVMS Cluster.

Note

The SYSTEST_CLIG account could have been disabled during the OpenVMS upgrade procedure. If it was disabled, you must reenable the SYSTEST_CLIG account and give it a null password before you run UETP.

To reenable the SYSTEST_CLIG account, enter the following commands:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF> MODIFY /FLAGS=NODISUSER  /NOPASSWORD SYSTEST_CLIG
UAF> EXIT
```

Note

VSI recommends that you disable the SYSTEST_CLIG account after testing has completed.

To disable the SYSTEST_CLIG account, enter the following commands:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF> MODIFY /FLAGS=DISUSER SYSTEST_CLIG
UAF> EXIT
```

- The privileges and quotas of the SYSTEST_CLIG account must match those of the SYSTEST account.

UETP requires little additional preparation for the cluster-integration test phase beyond the requirements for other UETP test phases. The additional requirements for cluster integration testing are as follows:

1. Your system must be a member of a cluster. If it is not, UETP displays a message and does not attempt to run the test.
2. Your system must use the same deadlock detection interval as the other systems in the cluster. (The deadlock detection interval is set by the system parameter DEADLOCK_WAIT. It is normally not changed from the default value, which is 10 seconds.)
3. The files UETCLIG00.COM and UETCLIG00.EXE, located in SYS$TEST, are necessary for each system included in the test.
4. DECnet for OpenVMS must be set up between the cluster nodes; UETP uses DECnet for OpenVMS to create a process on those nodes. All checks that the test makes depend on its ability to create the SYSTEST_CLIG processes and to communicate with them using DECnet for OpenVMS software.
5. All operator terminals (OPA0:) should accept broadcast messages. To set the BROADCAST characteristic, enter the following command:

   ```
   $ SET TERM/BROADCAST/PERM OPA0:
   ```

   Nodes on which the operator's terminal (OPA0) is set to the NOBROADCAST terminal characteristic will generate the following error message during the cluster test:

   ```
   **********************
   *  UETCLIG00master   *
   *  Error count =  1  *
   ```

```
**********************
-UETP-E-TEXT, 0 operator consoles timed out on the cluster test warning
       and 1 operator console rejected it.
-UETP-E-TEXT, Status returned was,
      "%SYSTEM-F-DEVOFFLINE, device is not in configuration or not
      available"
```

6. There must be a [SYSTEST] or [SYS0.SYSTEST] directory on some disk available to the cluster for each node (both OpenVMS and HSC) in the cluster. The test uses the same directory as the UETP disk test to create a file on each cluster node and to see if some other OpenVMS node in the cluster can share access to that file. There must be one such directory per node; the test continues with the next cluster node once it has finished with a file.

7. By default, the UETP cluster phase selects three nodes from the running cluster for deadlock, disk, and file access testing. However, if you want all cluster nodes tested, enter the following command before invoking UETP:

   ```
   $ DEFINE/GROUP UETP$CTMODE ALL
   ```

5.3.17. Testing a Small-Disk System

After you install the OpenVMS operating system on a small system disk (for example, an RZ23L), you might not have the 1200 blocks of free disk space required to run UETP successfully. If you do not have 1200 free blocks on your system disk, use VMSTAILOR to remove some files from the system disk before you run UETP. For instructions on using VMSTAILOR, refer to the *VSI OpenVMS Upgrade and Installation Manual* for your system.

5.3.18. DECnet for OpenVMS Phase

The DECnet for OpenVMS phase of UETP uses more system resources than other tests. You can, however, minimize disruptions to other users by running the test on the least busy node.

By default, the file UETDNET00.COM specifies the node from which the DECnet test will be run. To run the DECnet test on a different node, enter the following command before you invoke UETP:

```
$ DEFINE/GROUP UETP$NODE_ADDRESS node_address
```

This command equates the group logical name UETP$NODE_ADDRESS to the node address of the node in your area on which you want to run the DECnet phase of UETP.

For example:

```
$ DEFINE/GROUP UETP$NODE_ADDRESS 9.999
```

You can also run the DECnet for OpenVMS test on a different node by entering the following command before you invoke UETP:

```
$ DEFINE/GROUP UETP$NODE_NAME "node""username password""
```

Note

When you use the logical name UETP$NODE_ADDRESS, UETP tests only the first active circuit found by NCP (Network Control Program). Otherwise, UETP tests all active testable circuits.

When you run UETP, a router node attempts to establish a connection between your node and the node defined by UETP$NODE_ADDRESS or UETP$NODE_NAME. Occasionally, the connection between

your node and the router node can be busy or nonexistent. When this happens, the system displays the following error messages:

```
%NCP-F-CONNEC, Unable to connect to listener
-SYSTEM-F-REMRSRC, resources at the remote node were insufficient

%NCP-F-CONNEC, Unable to connect to listener
-SYSTEM-F-NOSUCHNODE, remote node is unknown
```

5.3.19. Vector Processors and the VVIEF (VAX Only)

UETP automatically loads all installed and enabled vector processors during the load phase, and automatically tests all installed and enabled vector processors during the device test phase.

If vector processors are available on the system, check for the VP number by entering the following commands:

```
$ x = F$GETSYI ("VP_NUMBER")
$ SHOW SYMBOL x
```

Multiply the value of *x* by 3. If the result is greater than the account PRCLM value, then you must increase the SYSTEST account PRCLM quota to match the returned result. For more information, see *Chapter 14, "Managing Special Processing Environments"*.

However, UETP cannot load the VAX Vector Instruction Emulation facility (VVIEF) during the load phase, and will not automatically test VVIEF. To test VVIEF, you must perform the following steps before running UETP:

1. Edit the file UETCONT00.DAT to add the following line:

   ```
   Y   Y  UETVECTOR.EXE  "DEVICE_TEST"
   ```

2. Make sure VVIEF was activated when the system was booted. To determine if the VVIEF was activated, enter the following DCL commands:

   ```
   $ X = F$GETSYI("VECTOR_EMULATOR")
   $ SHOW SYMBOL X
   ```

   If the system displays a value of 1, VVIEF is loaded;if the system displays a value of 0, VVIEF is not loaded.

The VVIEF test can be executed as an individual test using the RUN command, as described in *Section 5.8.2, "Device Test Phase"*.

5.4. Starting UETP

When you have logged in and prepared the system and devices, you are ready to begin the test.

To start UETP, enter the following command and press Return:

```
$ @UETP
```

UETP displays the following prompt:

```
Run "ALL" UETP phases or a "SUBSET" [ALL]?
```

Throughout the startup dialog, brackets indicate the default value, which you can choose by pressing Return.

When running UETP for the first time, it is recommended that you choose the default value (ALL) and run all the phases. If you choose ALL, UETP displays three more questions, which are described in *Section 5.4.2, "Single Run Versus Multiple Passes"* through *Section 5.4.4, "Report Formats"*. If you want to run all the test phases, skip the next section.

5.4.1. Running a Subset of Phases

You can run a single phase by entering SUBSET or S in response to the following prompt:

```
Run "ALL" UETP phases or a "SUBSET" [ALL]?
```

If you enter S or SUBSET, UETP prompts you for the phase you want to run as follows:

```
You can choose one or more of the following phases:

        DEVICE, LOAD, DECNET, CLUSTER

Phases(s):
```

There is no default; enter one or more phase names from the list. Separate two or more phases with spaces or commas.

If your choice includes the LOAD phase, UETP displays three prompts:

```
How many passes of UETP do you wish to run [1]?
How many simulated user loads do you want [n]?
Do you want Long or Short report format [Long]?
```

If you exclude the LOAD phase from your list of choices, UETP responds with only two prompts: the first and the third.

The next three sections discuss how you can respond to these questions. After you have answered the questions, the phase you have selected runs to completion.

5.4.2. Single Run Versus Multiple Passes

If you specified the default ALL or a subset of phases at the last prompt, UETP displays the following message:

```
How many passes of UETP do you wish to run [1]?
```

You can repeat the test run as many times as you want. If you enter 1 in response to the prompt (or press Return for the default), UETP stops after completing a single run. If you specify a number greater than 1, UETP restarts itself until it completes the specified number of passes.

You can run UETP once to check that the system is working, or many times to evaluate the system's response to continuous use. For example, a service technician who is interested only in verifying that a newly installed system works might run UETP once or twice. A manufacturing technician might let the system run for several hours as part of the system integration and test.

When you specify multiple UETP runs, you can request a short console log. (See *Section 5.4.4, "Report Formats"*.) Ensure that all line printers and hardcopy terminals have enough paper because each run requires two pages.

5.4.3. Defining User Load for Load Test

After you specify the number of passes, UETP prompts you as follows:

```
How many simulated user loads do you want [n]?
```

Note

UETP displays this prompt only if you choose to run the LOAD phase, either implicitly (by running all phases) or explicitly (by running a subset and specifying the LOAD phase).

The load test simulates a situation in which a number of users (detached processes) are competing for system resources. In response to this prompt, enter the number of users you want to simulate for this test. The number in brackets is the default value that UETP computed for your system. The default value depends on the amount of memory and the paging and swapping space that your system has allocated.

Although the given default value is the best choice, you can increase or decrease the user load by entering your own response to the prompt. However, be aware that an increase can cause the test to fail because of insufficient resources.

If you want to see UETP display the user-load equation as it runs, see *Section 5.6.2, "Interpreting UETP Output"*.

5.4.4. Report Formats

The following prompt allows you to choose between long or short report formats:

```
Do you want Long or Short report format [Long]?
```

5.4.4.1. Long Report Format

If you choose the long report format (the default), UETP sends the following information to the console terminal:

- All error messages
- All output generated at the beginning of all phases and tests
- All output generated at the end of all phases and tests

UETP records all its output in the UETP.LOG file, regardless of your response to this question.

In many cases, it might not be convenient to have UETP write the bulk of its output to the terminal. For example, if you run UETP from a hardcopy terminal, the output printing can slow the progress of the tests. This delay might not be a problem if you have requested only one run; however, you might prefer to use the short format if you intend to run multiple passes of UETP from a hardcopy terminal.

5.4.4.2. Short Report Format

If you request the short format, UETP displays status information at the console, such as error messages and notifications of the beginning and end of each phase. This information enables you to determine whether UETP is proceeding normally. If the short console log indicates a problem, you can look at the file UETP.LOG for further information. UETP.LOG contains all the output generated by the various phases, as well as the status information displayed at the console.

After you choose the report format, UETP initiates its sequence of tests and runs to completion. If UETP does not complete successfully, refer to *Section 5.6, "Troubleshooting: An Overview"* for troubleshooting information.

5.5. Stopping a UETP Operation

At the end of a UETP pass, the master command procedure UETP.COM displays the time at which the pass ended. In addition, UETP.COM determines whether UETP needs to be restarted. You can request multiple passes when you start up the test package. (See *Section 5.4.2, "Single Run Versus Multiple Passes"*.)

At the end of an entire UETP run, UETP.COM deletes temporary files and does other cleanup activities.

Pressing **Ctrl/Y** or **Ctrl/C** lets you terminate a UETP run before it completes normally. Normal completion of a UETP run, however, includes the deletion of miscellaneous files that have been created by UETP for the purpose of testing. Using **Ctrl/Y** or **Ctrl/C** can interrupt or prevent these cleanup procedures.

The effect of these control characters depends on what part of UETP you are executing. For an explanation of the organization of UETP and its components, refer to *Section 5.8, "UETP Tests and Phases"*.

5.5.1. Using Ctrl/Y

Press **Ctrl/Y** to abort a UETP run. Note, however, that cleanup of files and network processes in the [SYSTEST] directory might not be complete.

If you are running an individual test image, pressing **Ctrl/Y** interrupts the current UETP test and temporarily returns control to the command interpreter. While the test is interrupted, you can enter a subset of DCL commands that are executed within the command interpreter and do not cause the current image to exit.

5.5.2. Using DCL Commands

The *VSI OpenVMS User's Manual* contains a table of commands that you can use within the command interpreter. In addition, you can enter any of the following commands:

- The CONTINUE command continues the test from the point of interruption (except during execution of the cluster test).
- The STOP command terminates the test; the test aborts and control returns to the command interpreter.

  Note

  Using the STOP command can prevent cleanup procedures from executing normally. You should use the EXIT command if you want the image to do cleanup procedures before terminating.

- The EXIT command executes cleanup procedures and terminates the test (except during execution of the cluster test); control returns to the command interpreter.

If you enter any DCL command other than those that execute within the command interpreter, the test does cleanup procedures and terminates, and the DCL command executes.

5.5.3. Using Ctrl/C

Press **Ctrl/C** to interrupt a UETP run. You cannot continue the same test phase after you press **Ctrl/C**. UETP automatically goes to the next phase in the master command procedure.

Some UETP phases react to **Ctrl/C** by cleaning up all activity and terminating immediately. These tests display the following message when they are started:

```
%UETP-I-ABORTC, 'testname' to abort this test, type ^C
```

The phases that do not display the previous message terminate all processes they have started. These processes might not have a chance to complete normal cleanup procedures.

If you are running an individual test image, however, you can use **Ctrl/C** to terminate the execution of the image and complete cleanup procedures.

Note that **Ctrl/C** does not complete cleanup procedures for the cluster test.

5.6. Troubleshooting: An Overview

This section explains the role of UETP in interpreting operational errors in an OpenVMS operating system. See *Section 5.7, "Troubleshooting: Possible UETP Errors"* for a discussion of common errors that can appear in a UETP run and describes how to correct them.

5.6.1. Error Logging and Diagnostics

When UETP encounters an error, it reacts like a user program. It either returns an error message and continues, or it reports a fatal error and terminates the image or phase. In either case, UETP assumes the hardware is operating properly and it does not attempt to diagnose the error.

If the cause of an error is not readily apparent, use the following methods to diagnose the error:

- OpenVMS Error Log utility (ERROR LOG)—Run ERROR LOG to obtain a detailed report of hardware and system errors. ERROR LOG reports provide information about the state of the hardware device and I/O request at the time of each error. For information about running ERROR LOG refer to the *VSI OpenVMS System Management Utilities Reference Manual*.
- Diagnostic facilities—Use the diagnostic facilities to test exhaustively a device or medium to isolate the source of the error.

5.6.2. Interpreting UETP Output

You can monitor the progress of UETP tests at the terminal from which they were started. This terminal always displays status information, such as messages that announce the beginning and end of each phase and messages that signal an error.

The tests send other types of output to various log files, depending on how you started the tests. (See *Section 5.6.7, "Log Files"*.) The log files contain output generated by the test procedures. Even if UETP completes successfully, with no errors displayed at the terminal, it is good practice to check these log files for errors. Furthermore, when errors are displayed at the terminal, check the log files for more information about their origin and nature.

Each test returns a final completion status to the test controller image, UETPHAS00, using a termination mailbox. This completion status is an unsigned longword integer denoting a condition value. As a

troubleshooting aid, UETPHAS00 displays the test's final completion status using the $FAO and $GETMSG system services.

Sometimes, however, the $FAO service needs additional information that cannot be provided using the termination mailbox. When this happens, UETP displays an error message similar to the following one:

```
UETP-E-ABORT, !AS aborted at !%D
```

When UETP displays these types of error messages, check the log files for more information. You can also run the individual test to attempt to diagnose the problem.

The error messages that appear at the terminal and within the log files have two basic sources:

- UETP tests
- System components that are tested

If you need help interpreting the messages, use the OpenVMS Help Message utility (Help Message) or refer either to the *OpenVMS System Messages and Recovery Procedures Reference Manual* or to the manual that describes the individual system component.

5.6.3. Displaying Information on Your Screen

Several parts of UETP, such as some device tests, UETINIT00.EXE, UETCLIG00.EXE, and UETDNET00.COM, let you obtain additional information concerning the progress of the test run or the problems the test encounters. Because this information is usually insignificant, it is not displayed on the screen.

To view the information, enter the following command to define the logical name MODE and run the program:

```
$ DEFINE MODE DUMP
```

5.6.4. Example Screen Display (VAX Only)

The following example shows the output for UETINIT00.EXE on a VAX 6000 computer, and logical name MODE defined as DUMP:

```
$ DEFINE MODE DUMP
$ RUN UETINIT00

       Welcome to VAX/VMS UETP Version X7.3

%UETP-I-ABORTC, UETINIT00 to abort this test, type ^C

You are running on a VAX 6000-430 CPU with 327680 pages of memory.
The system was booted from _$11$DUA6:[SYS0.].

Run "ALL" UETP phases or a "SUBSET" [ALL]?
How many passes of UETP do you wish to run [1]?

The default number of loads is the minimum result of

1) CPU_SCALE * ((MEM_FREE + MEM_MODIFY) / (WS_SIZE * PER_WS_INUSE))
        7.32 * ((  232390 +       5048) / (   1024 *         0.20))  = 8486
```

```
2) Free process slots                                                = 296

3) Free page file pages / Typical use of page file pages per process
                1099992 /                                       1000 = 1099

How many simulated user loads do you want [296]?
Do you want Long or Short report format [Long]?

UETP starting at  1-MAR-2017 16:00:43.86 with parameters:
DEVICE LOAD DECNET CLUSTER  phases, 1 pass, 296 loads, long report.

$
```

This program does not initiate any phase; it displays the equation used by UETP to determine user load and the specific factors that are employed in the current run.

Respond to the questions by pressing Return. After you respond to the first prompt, the program displays the expressions that determine the default number of simultaneous processes. The following definitions apply:

- CPU_SCALE refers to the relative processing power of the CPU in relation to a VAX 11/780 computer. For example, a VAX 6000-430 computer has a CPU_SCALE of 7.32 because it has 7.32 times the processing power of a VAX 11/780 (1.0) computer.
- MEM_FREE represents memory in pages available to users.
- MEM_MODIFY represents memory pages on the modified page list.
- WS_SIZE represents working set size.
- PER_WS_INUSE represents typical percentage of the working set in active use for each process.

UETINIT00 also displays the specific values represented by the expressions. In this example, UETP selects 296 as the default for simulated user loads, because 296 is the minimum result of the three expressions.

Deassign the logical name MODE before running UETP, unless you prefer to see the user load breakdown every time you run UETP.

5.6.5. Example Screen Display (Alpha Only)

The following example shows the output for UETINIT00.EXE on an Alpha system, with logical name MODE defined as DUMP:

```
$ DEFINE MODE DUMP
$ RUN UETINIT00

       Welcome to OpenVMS Alpha UETP Version 7.3

%UETP-I-ABORTC, UETINIT00 to abort this test, type ^C

You are running on a AlphaServer 4100 5/533 4MB CPU.
The system was booted from _$4$DKA300:[SYS0.].

Run "ALL" UETP phases or a "SUBSET" [ALL]?
How many passes of UETP do you wish to run [1]?
```

```
The default number of loads is the minimum result of

1) (MEM_FREE + MEM_MODIFY) / ( WS_SIZE )
   ( 1807872 +      10496) / (  16512)                          = 110

2) Free process slots                                          = 488

3) Free page file pages / Typical use of blocks per process
                 650240 /                                1000 = 650

How many simulated user loads do you want [110]?
Do you want Long or Short report format [Long]?

UETP starting at  1-MAR-2017 15:53:19.52 with parameters:
DEVICE LOAD DECNET CLUSTER  phases, 1 pass, 110 loads, long report.
```

This program does not initiate any phase; it displays the equation used by UETP to determine user load and the specific factors that are employed in the current run.

Respond to the questions by pressing the Return key. After you respond to the first prompt, the program displays the expressions that determine the default number of simultaneous processes. The following definitions apply:

- MEM_FREE represents memory in pagelets available to users.
- MEM_MODIFY represents memory pagelets on the modified page list.
- WS_SIZE represents working set size in pagelets.

UETINIT00 also displays the specific values represented by the expressions. In this example, UETP selects 110 as the default for simulated user loads, because 100 is the minimum result of the three expressions.

Deassign the logical name MODE before running UETP, unless you prefer to see the user load breakdown every time you run UETP.

5.6.6. Defining a Remote Node for UETP Ethernet Testing

Occasionally during the UETUNAS00 test, it is difficult to determine whether the problem reports concern the device under test or the remote device. The easiest way to ensure proper error reporting is to define a *good turnaround.* A good turnaround is a remote node that you know turns around Ethernet packets correctly and is up and waiting in the ready state.

You can make the UETUNAS00 test use a known good turnaround by performing the following actions. In the commands that follow, assume that the *good* device is on node BETA and that node BETA is already defined in the network database.

1. Find the address of the good Ethernet node by using the Network Control Program (NCP). To use NCP, the following conditions must apply:
   - DECnet for OpenVMS must be up and running on the system.
   - The account you are using must have TMPMBX and NETMBX privileges.

   Enter the following commands and press Return:

```
$ RUN SYS$SYSTEM:NCP
NCP> TELL BETA SHOW EXECUTOR STATUS
```

If node BETA has not been defined in your network database, NCP displays an error message. In this event, specify another good node and retry the command. Otherwise, see your system or network manager.

NCP displays information similar to the following messages:

```
Node Volatile Status as of  22-JUN-2016 16:13:02

Executor node = 19.007 (BETA)

State                     = on
Physical address          = AA-00-03-00-76-D3
Active links              = 6
Delay                     = 1
```

2. Use the displayed `physical address` (in this case, AA00030076D3) to define the logical name TESTNIADR to point to the good turnaround. Note that you do not specify the hyphens (-).

   First, log in to the SYSTEST account. Then enter the following command:

   ```
   $ DEFINE/SYSTEM TESTNIADR AA00030076D3
   ```

3. Run UETP.

4. When UETP has completed, deassign the logical name TESTNIADR by entering the following command:

   ```
   $ DEASSIGN/SYSTEM TESTNIADR
   ```

5.6.7. Log Files

UETP stores all information generated by all UETP tests and phases from its current run in one or more UETP.LOG files, and it stores the information from the previous run in one or more OLDUETP.LOG files. If a run of UETP involves multiple passes, there will be one UETP.LOG or one OLDUETP.LOG file for each pass.

At the beginning of a run, UETP deletes all OLDUETP.LOG files, and renames any UETP.LOG files to equivalent versions of OLDUETP.LOG. Then UETP creates a new UETP.LOG file and stores the information from the current pass in it. Subsequent passes of UETP create higher versions of UETP.LOG. Therefore, at the end of a run of UETP that involves multiple passes, there is one UETP.LOG file for each pass. In producing the files UETP.LOG and OLDUETP.LOG, UETP provides the output from the two most recent runs.

The cluster test creates a NETSERVER.LOG file in SYS$TEST for each pass on each system included in the run. If the test is unable to report errors (for example, if the connection to another node is lost), the NETSERVER.LOG file on that node contains the result of the test run on that node. UETP does not purge or delete NETSERVER.LOG files; therefore, you must delete them occasionally to recover disk space.

If a UETP run does not complete normally, SYS$TEST can contain other log files. Ordinarily these log files are concatenated and placed within UETP.LOG. You can use any log files that appear on the system disk for error checking, but you must delete these log files before you run any new tests. You can delete

these log files yourself or rerun the entire UETP, which checks for old UETP.LOG files and deletes them.

5.7. Troubleshooting: Possible UETP Errors

This section is intended to help you identify and solve problems you can encounter running UETP. You should refer to this section if you need help understanding a system failure and isolating its cause. This section is not intended as a repair manual and is not expected to diagnose any flaws in your system. It should, however, help you to interpret and act upon the information in the error messages.

If you are unable to correct an error after following the steps in this section, you should contact a VSI support representative. Any information you can supply about the measures you have taken to isolate the problem will help your a VSI support representative diagnose the problem.

5.7.1. Summary of Common Failures

The following problems are the most common failures encountered while running UETP:

- Wrong quotas, privileges, or account
- UETINIT01 failure
- UETVECTOR failure (VAX computers only)
- Ethernet device allocated or in use by another application
- Insufficient disk space
- Incorrect cluster setup
- Problems during the load test
- DECnet for OpenVMS error
- Errors logged but not displayed
- No process control block (PCB) or swap slots
- System hangups
- Lack of default access for the file access listener (FAL) object
- Bug checks and machine checks

The sections that follow describe these errors and offer the best course of action for dealing with each one.

5.7.2. Wrong Quotas, Privileges, or Account

If your assigned quotas or privileges do not match standard quotas and privileges for the SYSTEST account, UETP displays the following error message:

```
**********************
*  UETINIT00          *
*  Error count =  1   *
```

```
**********************
-UETP-W-TEXT,  The following:

        OPER privilege,
        BIOLM  quota,
        ENQLM  quota,
        FILLM  quota,

are nonstandard for the SYSTEST account and may result in UETP errors.
```

This message informs you that the OPER privilege and the BIOLM, ENQLM, and FILLM quotas either are not assigned correctly or are not assigned at all.

Note

UETP displays a similar message if you run the cluster integration test phase and the privileges and quotas for the SYSTEST_CLIG account are incorrect. The SYSTEST and SYSTEST_CLIG accounts require the same privileges and quotas. Take the action described in this section for both accounts.

Solution

To correct the problem, use the following procedure:

1. Display all privileges and quotas in effect for the SYSTEST account using the Authorize utility (AUTHORIZE) as follows:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN SYS$SYSTEM:AUTHORIZE
UAF> SHOW SYSTEST

Username: SYSTEST                          Owner:  SYSTEST-UETP
Account:  SYSTEST                          UIC:    [1, 7] ([SYSTEST])
CLI:      DCL                              Tables: DCLTABLES
Default:  SYS$SYSROOT:[SYSTEST]
LGICMD:   LOGIN
Login Flags:
Primary days:   Mon Tue Wed Thu Fri Sat Sun
Secondary days:
No access restrictions
Expiration:            (none)    Pwdminimum:  8   Login Fails:     0
Pwdlifetime:         14 00:00    Pwdchange:   22-JUN-2016 10:12
Last Login:            (none) (interactive),         (none) (non-
interactive)
Maxjobs:         0  Fillm:       100  Bytlm:        65536
Maxacctjobs:     0  Shrfillm:      0  Pbytlm:           0
Maxdetach:       0  BIOlm:        12  JTquota:       1024
Prclm:          12  DIOlm:        55  WSdef:          256
Prio:            4  ASTlm:       100  WSquo:          512
Queprio:         0  TQElm:        20  WSextent:      2048
CPU:        (none)  Enqlm:       300  Pgflquo:      20480
Authorized Privileges:
  CMKRNL CMEXEC SYSNAM GRPNAM DETACH DIAGNOSE LOG_IO GROUP
  PRMCEB PRMMBX SETPRV TMPMBX NETMBX VOLPRO PHY_IO SYSPRV
Default Privileges:
  CMKRNL CMEXEC SYSNAM GRPNAM DETACH DIAGNOSE LOG_IO GROUP
  PRMCEB PRMMBX SETPRV TMPMBX NETMBX VOLPRO PHY_IO SYSPRV
UAF> SHOW SYSTEST_CLIG
```

```
⋮
UAF> EXIT
```

2. Make sure the default privileges and quotas assigned to the account match the following list:

   Privileges

CMKRNL	CMEXEC	NETMBX	DIAGNOSE
DETACH	PRMCEB	PRMMBX	PHY_IO
GRPNAM	TMPMBX	VOLPRO	LOG_IO
SYSNAM	SYSPRV	SETPRV	GROUP

   Quotas

BIOLM: 18	PRCLM: 12
DIOLM: 55	ASTLM: 100
FILLM: 100	BYTLM: 65536
TQELM: 20	CPU: no limit
ENQLM: 300	PGFLQUOTA: 20480
WSDEFAULT: 256	WSQUOTA: 512
WSEXTENT: 2048	

3. If any privileges or quotas are incorrect, run AUTHORIZE to correct them.

If you are logged in to the wrong account, the following error message asks you to log in to the SYSTEST account:

```
$ @UETP

**********************
*  UETINIT00         *
*  Error count =  1  *
**********************
-UETP-E-ABORT, UETINIT00 aborted at  22-JUN-2016 14:24:10.13
-UETP-E-TEXT, You are logged in to the wrong account.
              Please log in to the SYSTEST account.
$
```

You must run UETP from the SYSTEST account.

5.7.3. UETINIT01 Failure

UETINIT01 failures are related to peripheral devices; this type of error message can indicate any of the following problems:

- Device failure
- Device not supported or not mounted
- Device allocated to another user
- Device write locked

- Lost vacuum on a magnetic tape drive
- Drive off line

In some cases, the corrective action is specified explicitly in the error message. For example, you can receive a message from the operator communication manager (OPCOM) informing you of a problem and recommending a corrective measure:

```
%OPCOM, 22-JUN-2016 14:10:52.96, request 1, from user SYSTEST
Please mount volume UETP in device _MTA0:
%MOUNT-I-OPRQST, Please mount volume UETP in device _MTA0:
```

Other error messages can relate information in which the solution is specified implicitly:

```
%UETP-S-BEGIN, UETDISK00 beginning at 22-JUN-2016 13:34:46.03

**********************
*  DISK_DRA          *
*  Error count =  1  *
**********************
-UETP-E-TEXT, RMS file error in file DRA0:DRA00.TST
-RMS-E-DNR, device not ready or not mounted
%UETP-S-ENDED, UETDISK00 ended at  22-JUN-2016 13:34:46.80
```

This message tells you that a disk drive is either not ready or not mounted. From this information, you know where to look for the cause of the failure (at the disk drive). If you cannot see the cause of the problem immediately, check the setup instructions in *Section 5.3, "Setting Up the Devices to Be Tested"*.

In other cases, the cause of a failure might not be obvious from the information in the message. The problem can be related to hardware rather than software. For example, the Ethernet adapter test may produce one of the following messages if UETP does not have exclusive access to the Ethernet adapter:

- Intermodule cable unplugged
- Self-test failure code 0000000

To run the self-test diagnostic on the Ethernet adapter successfully, UETP needs exclusive access to the adapter. As explained in *Section 5.3.10, "Ethernet Adapters"*, you must shut down DECnet and the LAT terminal server before running the UETP device test phase if you want to test the Ethernet adapter.

Solution

To determine where or when the failure occurs in the execution of UETP, use the following procedure:

- Run the device test individually. (See *Section 5.4.1, "Running a Subset of Phases"*.) By doing this, you can determine if the failure can be re-created, and you can isolate the cause of the problem by reproducing it using the least amount of software possible.

  For example, if the failure occurs only when you run the entire device phase, and not when you run the affected device test individually, you can conclude the problem is related to device interaction. Conversely, if you can re-create the error by running the single device test, then you have proved that the error is not related to device interaction.

- Run the device test with different media. If your run of the single device test succeeded in reproducing the error, the magnetic tape or disk media could be defective. Running the same test with different media determines whether the original media caused the problem.

- Call a VSI support representative. If you have tried all the previous steps without solving the problem, you should contact a VSI support representative.

5.7.4. UETVECTOR Failure (VAX Only)

UETP displays a message similar to the following one to signal a vector processor failure:

```
     **********************
     *  UETVECTOR         *
     *  Error count = 1   *
     **********************
     %PPL-S-CREATED_SOME, created some of those requested - partial success
     -UETP-E-SUBSPNERR, Error spawning subordinate process.
     -UETP-E-SCHCTXERR, Error scheduling vector context test subprocess.
     -UETP-E-VECCTXERR, Error encountered during vector context testing.
%UETP-I-ENDED, UETVECTOR_0000 ended at 22-JUN-2016 07:37:00.59
```

Solution

See *Section 5.3.19, "Vector Processors and the VVIEF(VAX Only)"* for the correct setup for vector processor testing.

5.7.5. Device Allocated or in Use by Another Application

If DECnet for OpenVMS software or the LAT software is running during the DEVICE phase, the UETUNAS00 test displays the following message:

```
-UETP-W-TEXT, Device is in use by DECnet or another application
```

Other UETP communication device tests display the following message:

```
SYSTEM-W-DEVALLOC, device already allocated to another user
```

Solution

If you want to run the device test on the Ethernet adapter, shut down DECnet and LAT software before beginning the test.

5.7.6. Insufficient Disk Space

When you run continuous passes of UETP, log files accumulate on the disk from which UETP was run. These files reduce the amount of free disk space available for each successive pass. If the amount of disk space available becomes too small for the current load, the following error message appears:

```
%UETP-S-BEGIN, UETDISK00 beginning at  22-JUN-2016 08:12:24.34
%UETP-I-ABORTC, DISK_DJA to abort this test, type ^C

**********************
*  DISK_DJA          *
*  Error count = 1   *
**********************
-UETP-F-TEXT, RMS file error in file DJA0:DJA00.TST
```

```
-RMS-F-FUL, device full (insufficient space for allocation)

**********************
*  DISK_DJA          *
*  Error count = 2   *
**********************
-UETP-F-TEXT, RMS file error in file DJA0:DJA01.TST
-RMS-F-FUL, device full (insufficient space for allocation)
%UETP-E-DESTP, DISK_DJA stopped testing DJA unit 0 at 08:12:36.91
%UETP-S-ENDED, UETDISK00 ended at  22-JUN-2016 08:12:37.98
```

Solution

Make more space available on the disk. You can do this by using one or more of the following techniques:

- Delete unnecessary files to create more space.
- Purge files, if multiple versions exist.
- Mount a volume with sufficient space.
- Check for disk quotas that might be enabled on the disk. If disk quotas are enabled, either disable or increase them. (Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for a description of the Disk Quota utility.)
- Run VMSTAILOR if you have a small-disk system. Refer to the upgrade and installation manual for your operating system for more information.

See *Section 5.2.2, "Using the SYSTEST Directories"* and *Section 5.3.3, "How UETP Works on Disks"* for a further discussion of disk space.

5.7.7. Incorrect Setup of an OpenVMS Cluster System

Most problems that can occur during the cluster-integration test are related to improper setup of the OpenVMS Cluster system or of UETP on the cluster. These problems are most likely to occur at the following stages of the cluster test:

- Near the beginning, when processes on OpenVMS nodes are started
- Toward the end, when cluster file access is checked

The cluster test phase shows that various OpenVMS nodes in your cluster can simultaneously access files on selected nodes in the cluster. First, UETP tries to create a file on a disk drive that is accessible to the other selected nodes in the cluster. The following requirements are for creating a file in the cluster test phase:

- A [SYSTEST] directory must exist on the disk in either the master file directory (MFD) or in the root directory [SYS0.].
- The protection for [SYSTEST] directory must be set to allow the SYSTEST account to create a file in it.

If UETP is unable to find a suitable device on a certain node, the test displays a warning message and proceeds to the next cluster node.

Nodes on which the operator's terminal (OPA0) is set to the NOBROADCAST terminal characteristic will generate the following error message during the cluster test:

```
**********************
*  UETCLIG00master   *
*  Error count =  1  *
**********************
-UETP-E-TEXT, 0 operator consoles timed out on the cluster test warning
       and 1 operator console rejected it.
-UETP-E-TEXT, Status returned was,
      "%SYSTEM-F-DEVOFFLINE, device is not in configuration or not
      available"
```

Disregard this message if OPA0 is set to NO BROADCAST.

Solution

Whenever you suspect a problem, examine the SYS$TEST:NETSERVER.LOG file that was created when the SYSTEST_CLIG process was created. This file can contain additional error information that could not be transmitted to the node running the test. If it was not possible to create the SYSTEST_CLIG process on some node, the system accounting file for that node might contain a final process status in a process termination record.

The following problems can occur during a cluster test:

- Logging in at other nodes—This problem is due to incorrect setup for the cluster test at the remote OpenVMS node. For example, if you specified a password for the SYSTEST_CLIG account or if you disabled the SYSTEST_CLIG account, the test displays the following message:

  ```
  %SYSTEM-F-INVLOGIN, login information invalid at remote node
  ```

  Refer to *Section 5.3.16, "OpenVMS Cluster Testing"* and *Section 5.6.6, "Defining a Remote Node for UETP Ethernet Testing"* for information about preparing for cluster testing.

- Communicating with other nodes—A message indicates a DECnet problem. Check the NETSERVER.LOG file on the affected node to determine the cause.

- Taking out locks or detecting deadlocks—The most likely cause of this problem is that you are not logged in to the SYSTEST account. Another possibility is that your cluster is not configured properly.

- Creating files on cluster nodes—This problem is due to incorrect setup for the cluster test; refer to *Section 5.3.16, "OpenVMS Cluster Testing"* for information about preparing for cluster testing.

5.7.8. Problems During the Load Test

A variety of errors can occur during the load test because the command procedures that are started during the tests run several utilities and do many functions. Tracking a problem can be difficult because UETP deletes the log files that are generated during the load test. (See *Section 5.8.3, "System Load Test Phase"*.)

Solution

If a problem occurs during the load test and the cause is not obvious, you can modify UETP.COM to preserve the log files as follows:

1. Add the /NODELETE qualifier to the following line:

```
$ TCNTRL UETLOAD00.DAT/PARALLEL_COUNT='LOADS/REPORT_TYPE='REPORT
```

2. Delete or comment out the following line:

```
$ DELETE UETLO*.LOG;*
```

Rerun the load test with these changes to try to re-create the problem.

If you re-create the problem, look at the contents of the appropriate log file. You can determine which log file to read by understanding the scheme by which the load test names its processes and log files. (The log file names are derived from the process names.)

The load test creates processes that are named in the following format:

```
UETLOAD nn_nnnn
```

For example:

```
%UETP-I-BEGIN, UETLOAD00 beginning at 22-JUN-2016 15:45:08.97
%UETP-I-BEGIN, UETLOAD02_0000 beginning at 22-JUN-2016 15:45:09.42
%UETP-I-BEGIN, UETLOAD03_0001 beginning at 22-JUN-2016 15:45:09.63
%UETP-I-BEGIN, UETLOAD04_0002 beginning at 22-JUN-2016 15:45:10.76
%UETP-I-BEGIN, UETLOAD05_0003 beginning at 22-JUN-2016 15:45:11.28
%UETP-I-BEGIN, UETLOAD06_0004 beginning at 22-JUN-2016 15:45:12.56
%UETP-I-BEGIN, UETLOAD07_0005 beginning at 22-JUN-2016 15:45:13.81
%UETP-I-BEGIN, UETLOAD08_0006 beginning at 22-JUN-2016 15:45:14.95
%UETP-I-BEGIN, UETLOAD09_0007 beginning at 22-JUN-2016 15:45:16.99
%UETP-I-BEGIN, UETLOAD10_0008 beginning at 22-JUN-2016 15:45:19.32
%UETP-I-BEGIN, UETLOAD11_0009 beginning at 22-JUN-2016 15:45:19.95
%UETP-I-BEGIN, UETLOAD02_0010 beginning at 22-JUN-2016 15:45:20.20
%UETP-I-BEGIN, UETLOAD03_0011 beginning at 22-JUN-2016 15:45:21.95
%UETP-I-BEGIN, UETLOAD04_0012 beginning at 22-JUN-2016 15:45:22.99
```

Note that if more than 10 processes are created, the numbering sequence for the UETLOAD *nn* portion of the process name starts over at UETLOAD02; however, the 4 digits of the *_nnnn* portion continue to increase.

Each load test process creates two log files. The first log file is created by the test controller; the second log file is created by the process itself. The log file to look at for error information about any given load test process is the one that was created by the test controller (the first log file).

The load test log file derives its file name from the process name, appending the last four digits of the process name (from the _nnnn portion) to UETLO. The test-controller log file and the process log file for each process use the same file name; however, the process log file has the higher version number of the two. For example, the log files created by the process UETLOAD05_0003 would be named as follows:

```
UETLO0003.LOG;1 (test-controller log file)
UETLO0003.LOG;2 (process log file)
```

Make sure that you look at the log file with the lower version number;that file contains the load test commands and error information.

After you have isolated the problem, restore UETP.COM to its original state and delete the log files from the load test (UETL0*.LOG;*); failure to delete these files can result in disk space problems.

5.7.9. DECnet for OpenVMS Error

A DECnet error message can indicate that the network is unavailable.

Solution

- If DECnet for OpenVMS software is included in your system, determine whether the product authorization key (PAK) is registered by entering the following command:

  ```
  $ SHOW LICENSE
  ```

  If the PAK is not registered, invoke the License utility to register it by entering the following command:

  ```
  $ @SYS$UPDATE:VMSLICENSE
  ```

  For information about registering licenses, refer to the following documents:

  - The *VSI OpenVMS Upgrade and Installation Manual* for your operating system
  - The *VSI OpenVMS License Management Utility Guide*

- If DECnet for OpenVMS software is not included in your system, ignore the message; it is normal and does not affect the UETP run.

If you encounter other DECnet related errors, you should perform the following actions:

- Run DECnet for OpenVMS software as a single phase (see *Section 5.4.1, "Running a Subset of Phases"*) to determine whether the error can be re-created.
- Use the Help Message or refer to the *OpenVMS System Messages: Companion Guide for Help Message Users*.

5.7.10. Errors Logged but Not Displayed

If no errors are displayed at the console terminal or reported in the UETP.LOG file, you should run ERROR LOG to see if any errors were logged in the ERRLOG.SYS file. Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for information about running the ERROR LOG.

5.7.11. No PCB or Swap Slots

The following error message indicates that no PCB or swap slots are available:

```
%UETP-I-BEGIN, UETLOAD00 beginning at  22-JUN-2016 07:47:16.50
%UETP-I-BEGIN, UETLOAD02_0000 beginning at  22-JUN-2016 07:47:16.76
%UETP-I-BEGIN, UETLOAD03_0001 beginning at  22-JUN-2016 07:47:16.92
%UETP-I-BEGIN, UETLOAD04_0002 beginning at  22-JUN-2016 07:47:17.13
%UETP-I-BEGIN, UETLOAD05_0003 beginning at  22-JUN-2016 07:47:17.35
%UETP-I-BEGIN, UETLOAD06_0004 beginning at  22-JUN-2016 07:47:17.61
%UETP-W-TEXT, The process -UETLOAD07_0005- was unable to be created,
  the error message is
-SYSTEM-F-NOSLOT, no pcb or swap slot available
%UETP-W-TEXT, The process -UETLOAD08_0006- was unable to be created,
  the error message is
-SYSTEM-F-NOSLOT, no pcb or swap slot available
%UETP-W-TEXT, The process -UETLOAD09_0007- was unable to be created,
```

```
  the error message is
-SYSTEM-F-NOSLOT, no pcb or swap slot available
%UETP-W-TEXT, The process -UETLOAD10_0008- was unable to be created,
  the error message is
-SYSTEM-F-NOSLOT, no pcb or swap slot available
%UETP-W-TEXT, The process -UETLOAD11_0009- was unable to be created,
  the error message is
-SYSTEM-F-NOSLOT, no pcb or swap slot available
%UETP-W-ABORT, UETLOAD00 aborted at  22-JUN-2016 07:47:54.10
-UETP-W-TEXT, Aborted via a user Ctrl/C.
 ************************************************
 *                                              *
    END OF UETP PASS 1 AT  22-JUN-2016 07:48:03.17
 *                                              *
 ************************************************
```

Solution

To solve this problem, use the following procedure:

1. Individually rerun the phase that caused the error message (the LOAD phase in the previous example) to see if the error can be reproduced.
2. Increase the size of the page file, using either the command procedure SYS $UPDATE:SWAPFILES.COM (see *Chapter 2, "Managing Page, Swap, and Dump Files"*) or SYSGEN (refer to the *VSI OpenVMS System Management Utilities Reference Manual*).
3. Increase the system parameter MAXPROCESSCNT, if necessary.
4. Reboot the system.

5.7.12. No Keyboard Response or System Disk Activity

If the keyboard does not respond or the system disk is inactive, the system might be hung.

Solution

A system hangup can be difficult to trace; you should save the dump file for reference. To learn why the system hung, run the System Dump Analyzer as described in the *OpenVMS Alpha System Dump Analyzer Utility Manual* or the *VSI OpenVMS Alpha System Analysis Tools Manual*.

Reasons for a system hangup include the following ones:

- Insufficient pool space—Increase the value of the system parameter NPAGEVIR and reboot the system.
- Insufficient page file space—Increase the page file space using the SYSGEN as described in the *VSI OpenVMS System Management Utilities Reference Manual*.
- I/O device failure causing driver-permanent loop—Call a VSI support representative.

5.7.13. Lack of Default Access for the FAL Object

If default FAL access is disabled at the remote node selected by UETP for DECnet testing (the adjacent node on each active circuit, or a node defined by the group logical name UETP$NODE_ADDRESS), messages similar to the following ones appear:

```
%UETP-W-TEXT, The process -SVA019841_0001- returned a final status of:
%COPY-E-OPENOUT, error opening !AS as output
```

These messages are followed by:

```
%COPY-E-OPENOUT, error opening 9999"":::SVA019841.D1; as output
-RMS-E-CRE, ACP file create failed
-SYSTEM-F-INVLOGIN, login information invalid at remote node
%COPY-W-NOTCOPIED, SYS$COMMON:[SYSTEST]UETP.COM;2 not copied
%UETP-E-TEXT, Remote file test data error
```

You can ignore these messages.

5.7.14. Bugchecks and Machine Checks

When the system aborts its run, a bugcheck message appears at the console.

Solution

Call your VSI support representative. Often a hardware problem causes bug checks and machine checks; solving bug checks or machine checks is not easy. However, saving the SYS$SYSTEM:SYSDUMP.DMP and ERRLOG.SYS files is important so they are available for examination. Knowing whether the failure can be re-created is also important; you can run UETP again to verify the failure.

5.8. UETP Tests and Phases

This section explains, in detail, the organization of UETP and the individual components within the test package. You run UETP by starting a master command procedure containing commands to start each test phase. The procedure begins by prompting you for information needed by the various test phases. (See *Section 5.4, "Starting UETP"* for a detailed description of starting UETP.)

The master command procedure, UETP.COM, contains commands that initiate each test phase. UETP.COM also contains commands that do such tasks as defining logical names and manipulating files generated by the tests.

The UETP.COM procedure also issues commands to start the test controlling program UETPHAS00.EXE, which, in turn, controls each test phase. The test controller starts up multiple detached processes. It also reports their completion status and other information the processes report to it.

The sections that follow describe the various UETP test phases.

5.8.1. Initialization Phase

The following actions occur during the initialization phase:

- The image UETINIT00.EXE prompts you for information. (See *Section 5.4, "Starting UETP"*.) Your information defines variables that affect the execution of UETP tests.
- The image UETINIT01.EXE gathers information about all the controllers in the system and on their associated devices. This image writes the information into a file called UETINIDEV.DAT.
- Using the information in UETSUPDEV.DAT, UETINIT01.EXE verifies which devices in the configuration are operable by running the appropriate device test. Each device test completes

a simple read/write operation to each device. If a device fails this test, the device's entry in UETINIDEV.DAT specifies that the device cannot be tested. As a result, subsequent UETP tests ignore that device.

- For each testable controller, UETINIT01.EXE writes a line into a file called UETCONT00.DAT. The line associates a test file with the controller it tests.

A summary of UETINIDEV.DAT always exists in UETP.LOG, and UETINIT01.EXE sends that summary to the console if you have requested the long report format.

5.8.2. Device Test Phase

The device test phase includes separate tests for each type of device, such as disk, magnetic tape, line printer, and terminal. This section explains the device test phase and presents instructions for testing a single device. If you want to run the entire device test phase individually, refer to *Section 5.4.1, "Running a Subset of Phases"*.

5.8.2.1. How the Device Phase Works

The UETP device test phase starts an executable image, the phase controller UETPHAS00, which creates a detached process for every device controller to be tested. For example, if a system includes three terminal controllers, one line printer controller, and two disk controllers, the image creates six detached processes. In parallel, the detached processes execute images that test the various types of devices.

The initialization phase of UETP creates a file called UETINIDEV.DAT and a file called UETCONT00.DAT. UETINIDEV.DAT contains data on the controllers in the system supported by OpenVMS and their associated devices; UETCONT00.DAT associates a device test image with each testable controller.

UETPHAS00 uses the information in UETCONT00.DAT to find a device controller name to pass to each detached process that it creates. UETPHAS00 passes the controller name by writing it to a mailbox that is SYS$INPUT to individual tests. Each detached process uses that data to determine which controller to test. The test image then searches UETINIDEV.DAT for the device controller and for all testable units on that controller. The phase controller terminates when all devices on all controllers have completed testing.

Because UETCONT00.DAT is deleted automatically at the end of a UETP run, you cannot run the device phase unless you start UETP.COM; you can run only individual test images. UETINIDEV.DAT exists in SYS$TEST unless you delete it.

5.8.2.2. Running a Single Device Test

You must be logged in to the SYSTEST account to run the individual tests as described in this section. Also, a copy of UETINIDEV.DAT must exist. If a copy of the file is not present from a previous run (a run of the entire UETP or a run of the device test phase creates UETINIDEV.DAT), you can create it. Note that when you run a single test, no log file is created; the test sends all its output to your terminal.

If you do not want to test all the device types, you can test a specific controller by choosing a test image name from *Table 5.1, "Device Tests (VAX Only)"* (for VAX systems) or *Table 5.2, "Device Tests (Alpha Only)"* (for Alpha systems) and executing it as in the following example:

```
$ RUN UETTTYS00
```

```
Controller designation?: TTB
```

UETP prompts you for the controller designation and the device code. Unless you are testing your own terminal, you must explicitly designate a controller name. If you are running the terminal test, you can press Return to test your terminal only.

If you plan to repeat the run several times, you might find it more convenient to define the logical name CTRLNAME as follows:

```
$ DEFINE CTRLNAME TTB
$ RUN UETTTYS00
```

When you define the controller name in this way, the logical name CTRLNAME remains assigned after the test completes. To deassign this logical name, use the DCL command DEASSIGN as follows:

```
$ DEASSIGN CTRLNAME
```

5.8.2.3. Format of UETINIDEV.DAT

The UETINIDEV.DAT file is an ASCII sequential file that you can type or edit if necessary. The contents of this file are shown in the following command sequence:

```
$ TYPE UETINIDEV.DAT

DDB x ddd
UCB y uuuuu nnnnnnnnn.nnn
END OF UETINIDEV.DAT
```

The symbols in this example are defined as follows:

Symbol	Value
x	T, if testable units exist for this controller; N, if this controller is not to be tested
y	T, if this unit is testable; N, if this unit is not testable
ddd	Device controller name, for example DUA
uuuuu	Device unit number, for example 25
nnnnnnnnn.nnn	UETP device test name for the unit, for example, UETDISK00.EXE

UETINIDEV.DAT contains a DDB (device data block) line for each controller connected or visible to your system. After the DDB line is a UCB (unit control block) line for each unit connected to that controller. A device test can test a particular device only if both the DDB line and the UCB line indicate that the device is testable.

5.8.2.4. Running a Test in Loop Mode

If you want to put extra stress on a device, you can run the device test in loop mode, which causes the test to run indefinitely. For example:

```
$ DEFINE MODE LOOP
$ RUN UETDISK00
Controller designation?: DRA
%UETP-I-TEXT, End of pass 1 with 980 iterations at 22-JUN-2016 16:18:51:03

^C
```

You must use **Ctrl/C** to terminate the test run. If you use **Ctrl/Y**, UETP does not complete cleanup procedures.

5.8.2.5. Functions of Individual Device Tests

For each disk in the system, the disk test allocates two files into which it randomly writes blocks of data. The test then checks the data, reports any errors to SYS$OUTPUT, and deletes the disk files.

When you run the disk test phase in a cluster environment, the test accesses all disks that are mounted by the system being tested, and users of the disk being tested might encounter an insufficient disk space problem. You should warn users on remote nodes (who share disks with users on the local system) that UETP might be testing a disk they are using.

The magnetic tape test exercises all the magnetic tape drives in the system. The test creates a large file on each mounted magnetic tape, into which it writes multiple sequential records of varying sizes. After writing the records, the test rewinds the magnetic tape, validates the written records, and reinitializes the magnetic tape.

The terminal and line printer test generates several pages or screens of output, in which each page or screen contains a header line and a test pattern of ASCII characters. A header line contains the test name, the device name, the date, and the time.

For the laboratory peripheral accelerator (LPA11–K), the test image determines the configuration on the LPA11–K's I/O bus. The image loads all types of microcode to the LPA11–K and reads or writes data for each device on the LPA11–K I/O bus.

The communications device tests fill the transmit message buffer with random data; then, using loopback mode, the tests transmit and receive the message several times. To check that the looped-back data is correct, an AST routine is associated with a $QIO read to compare the received message against the transmitted message. The procedure is repeated using messages of different lengths.

The interface device tests put the devices they are testing in maintenance mode, write random data, and then verify the data.

The Ethernet adapter test does self-test diagnostics on the device. It also does read and write tasks with test data that uses various adapter modes (such as internal loopback and external loopback).

The vector processor device test performs simple vector-scalar and vector-vector arithmetic operations and compares the results with expected values. The test also uses vector-related system service extensions and forces the system to generate arithmetic and memory management exceptions.

Table 5.1, "Device Tests (VAX Only)" lists the device test images and the devices to be tested on VAX systems.

Table 5.1. Device Tests (VAX Only)

Test Image Name	Devices Tested
UETDISK00.EXE	Disks
UETTAPE00.EXE	Magnetic tape drives and tape cartridge drives
UETTTYS00.EXE	Terminals and line printers
UETLPAK00.EXE	LPA11–K
UETCOMS00.EXE	DMC11, DMR11
UETDMPF00.EXE	DMF32, DMP11

Test Image Name	Devices Tested
UETDR1W00.EXE	DR11–W
UETDR7800.EXE	DR780, DR750
UETCDRO00.EXE	RRD40, RRD42, RRD50
UETUNAS00.EXE	Ethernet Adapters
UETVECTOR.EXE	Vector Processor, VVIEF

Table 5.2, "Device Tests (Alpha Only)" lists the device test images and the devices to be tested on Alpha systems.

Table 5.2. Device Tests (Alpha Only)

Test Image Name	Devices Tested
UETDISK00.EXE	Disks
UETTAPE00.EXE	Magnetic tape drives and tape cartridge drives
UETTTYS00.EXE	Terminals and line printers
UETCDRO00.EXE	RRD42
UETUNAS00.EXE	Ethernet adapters

5.8.3. System Load Test Phase

The purpose of the system load test is to simulate a number of terminal users who are demanding system resources simultaneously. The system load tests, directed by the file UETLOAD00.DAT, create a number of detached processes that execute various command procedures. Each process simulates a user logged in at a terminal; the commands within each procedure are the same types of commands that a user enters from a terminal. The load test creates the detached processes in quick succession, and the processes generally execute their command procedures simultaneously. The effect on the system is analogous to an equal number of users concurrently issuing commands from terminals. In this way, the load test creates an environment that is similar to normal system use.

The load test uses the logical name LOADS to determine the number of detached processes to create. When you initiate the UETP command procedure, it prompts for the number of users to be simulated (see *Section 5.4.3, "Defining User Load for Load Test"*) and consequently the number of detached processes to be created. Your response, which depends on the amount of memory and the swapping and paging space in your system, defines the group logical name LOADS.

The UETP master command procedure deassigns all group logical names assigned by its tests as part of the termination phase. The group logical name LOADS remains assigned only if the UETP package does not complete normally.

The command procedures executed by the load test can generate a large amount of output, depending on the number of detached processes created. For each detached process (or user), the test creates a version of an output file called UETLO *nnnn*.LOG (*nnnn* represents a string of numeric characters). The console displays only status information as the load test progresses.

Whether the load test runs as part of the entire UETP or as an individual phase, UETP combines the UETLO nnnn.LOG files, writes the output to the file UETP.LOG, and deletes the individual output files.

You can run the system load test as a single phase by selecting LOAD from the choices offered in the startup dialog. (See *Section 5.4.1, "Running a Subset of Phases"*.)

5.8.4. DECnet for OpenVMS Test Phase

If DECnet for OpenVMS software is included in your OpenVMS system, a run of the entire UETP automatically tests DECnet hardware and software. Because communications devices are allocated to DECnet and the DECnet devices cannot be tested by the UETP device test, UETP will not test the Ethernet adapter if DECnet for OpenVMS or another application has allocated the device. The DECnet node and circuit counters are zeroed at the beginning of the DECnet test to allow for failure monitoring during the run.

As with other UETP phases, you can run the DECnet for OpenVMS phase individually by following the procedure described in *Section 5.4.1, "Running a Subset of Phases"*.

5.8.4.1. Environment

The DECnet for OpenVMS test will work successfully on OpenVMS systems connected to all DECnet supported node types, including routing and non routing nodes and several different types of operating systems (such as RSTS, RSX, TOPS, and RT). To copy files between systems, the remote systems must have some type of default access. The DECnet phase tests the following nodes and circuits:

- The node on which UETP is running.
- All circuits in sequence, unless you have defined the logical name UETP$NODE_ADDRESS to be the remote node that you want to run the test on. If you have defined a remote node, the DECnet phase tests only one circuit.
- All adjacent or first-hop nodes and all circuits in parallel.

No limit exists on the number of communication lines supported by the tests. A test on one adjacent node should last no more than two minutes at normal communications transfer rates.

Note

UETP assumes your system has default access for the FAL object, even though the network configuration command procedure NETCONFIG.COM does not provide access for the FAL object by default. When you install DECnet software with the defaults presented by NETCONFIG.COM, the UETP DECnet phase can produce error messages. You can ignore these error messages. See *Section 5.7.13, "Lack of Default Access for the FAL Object"* for more information.

5.8.4.2. How the DECnet Phase Works

UETP (under the control of UETPHAS00.EXE) reads the file UETDNET00.DAT and completes the following steps during the DECnet for OpenVMS phase:

1. Executes a set of Network Control Program (NCP) LOOP EXECUTOR commands to test the node on which UETP is running.
2. Uses NCP to execute the command SHOW ACTIVE CIRCUITS. The results are placed in UETININET.TMP, from which UETP creates the data file UETININET.DAT. The UETININET.TMP file contains the following information for any circuit in the ON state but not in transition:
   - Circuit name
   - Node address

- Node name (if one exists)

The UETININET.TMP file is used throughout the DECnet phase to determine which devices to test.

3. Uses the UETININET.TMP file to create an NCP command procedure for each testable circuit. Each command procedure contains a set of NCP commands to zero the circuit and node counters and to test the circuit and adjacent node by copying files back and forth.

Note

If you do not want the counters zeroed, do not test the DECnet for OpenVMS software.

---

4. Executes the command procedures from Step 3 in parallel to simulate a heavy user load. The simulated user load is the lesser of the following values:

   - The number of testable circuits, multiplied by two
   - The maximum number of user-detached processes that can be created on the system before it runs out of resources (determined by UETINIT00)

5. Executes a program, UETNETS00.EXE, that uses the UETININET.DAT file to check the circuit and node counters for each testable circuit. If a counter indicates possible degradation (by being nonzero), its name and value are reported to the console. All counters are reported in the log file, but only the counters that indicate degradation are reported to the console. An example of UETNETS00 output follows.

```
%UETP-S-BEGIN, UETNETS00 beginning at  22-JUN-2016 13:45:33.18
%UETP-W-TEXT, Circuit DMC-0 to (NODENAME1) OK.
%UETP-I-TEXT, Node  (NODENAME2) over DMC-1 response timeouts = 1.
%UETP-I-TEXT, Circuit DMC-1 to (NODENAME2) local buffer errors = 34.
%UETP-I-TEXT, Node  (NODENAME3) over DMP-0 response timeouts = 3.
%UETP-S-ENDED, UETNETS00 ended at 22-JUN-2016 13:45:36.34
```

   Because degradation is not necessarily an error, the test's success is determined by you, not by the system. The following counters indicate possible degradation:

   For Circuits

   - Arriving congestion loss
   - Corruption loss
   - Transit congestion loss
   - Line down
   - Initialization failure
   - Data errors inbound
   - Data errors outbound
   - Remote reply timeouts
   - Local reply timeouts

- Remote buffer errors
- Local buffer errors
- Selection timeouts
- Remote process errors
- Local process errors
- Locally initiated resets
- Network initiated resets

For Nodes

- Response timeouts
- Received connect resource errors
- Node unreachable packet loss
- Node out of range packet loss
- Oversized packet loss
- Packet format error
- Partial routing update loss
- Verification reject

5.8.5. Cluster-Integration Test Phase

The cluster-integration test phase consists of a single program and a command file that depend heavily on DECnet for OpenVMS software. This phase uses DECnet for OpenVMS software to create SYSTEST_CLIG processes on each OpenVMS node in the cluster and to communicate with each node. SYSTEST_CLIG is an account that is parallel to SYSTEST, but limited so that it can only be used as part of the cluster-integration test. The following restrictions on the SYSTEST_CLIG account are necessary for a correct run of the cluster test phase:

- The account must be enabled and the password must be null. For more information, see *Section 5.3.16, "OpenVMS Cluster Testing"*.
- The UIC must be the same as that of the SYSTEST account.
- The account must have the same privileges and quotas as the SYSTEST account. For more information, see *Section 5.7.2, "Wrong Quotas, Privileges, or Account"*.
- The account can allow login only through DECnet for OpenVMS software.
- The account must be locked into running UETCLIG00.COM when it logs in.

These items are necessary to ensure the security and privacy of your system. If the test cannot create a SYSTEST_CLIG process on an OpenVMS node, it gives the reason for the failure and ignores that node

for the lock tests and for sharing access during the file test. Also, the test does not copy log files from any node on which it cannot create the SYSTEST_CLIG process. If a communication problem occurs with a SYSTEST_CLIG process after the process has been created, the test excludes the process from further lock and file sharing tests. At the end of the cluster-integration test, an attempt is made to report any errors seen by that node.

UETCLIG00.EXE has two threads of execution: the primary and the secondary. The first, or primary thread, checks the cluster configuration (OpenVMS nodes, HSC nodes, and the attached disks that are available to the node running the test). For selected OpenVMS nodes, the primary thread attempts to start up a SYSTEST_CLIG process through DECnet software. If the primary thread was able to start a SYSTEST_CLIG process on a node, the node runs the command file UETCLIG00.COM, which starts up UETCLIG00.EXE and runs the secondary execution thread.

The process running the primary thread checks to see that it can communicate with the processes running the secondary threads. It then instructs them to take out locks so that a deadlock situation is created.

The primary thread tries to create a file on some disk on selected OpenVMS and HSC nodes in the cluster. It writes a block, reads it back, and verifies it. Next, it selects one OpenVMS node at random and asks that node to read the block and verify it. The primary thread then extends the file by writing another block and has the secondary thread read and verify the second block. The file is deleted.

The secondary processes exit. They copy the contents of their SYS$ERROR files to the primary process, so that the UETP log file and console reports how all problems in a central place. DECnet for OpenVMS software automatically creates a NETSERVER.LOG in SYS$TEST as the test is run, so that if necessary, you can read that file later from the node in question.

During the test run, the primary process uses the system service SYS$BRKTHRU to announce the beginning and ending of the test to each OpenVMS node's console terminal.

You can define the group logical name MODE to the equivalence string DUMP to trace most events as they occur. Note that the logical name definitions apply only to the node on which they were defined. You must define MODE on each node in the cluster on which you want to trace events.

Chapter 6. Getting Information About the System

This chapter discusses setting up and maintaining system log files, maintaining error log files, and using system management utilities to monitor the system.

This chapter describes the following tasks:

Task	Section
Using the Error Formatter (ERRFMT)	*Section 6.3, "Using the Error Formatter"*
Using ERROR LOG to produce reports	*Section 6.4, "Using the Error Log Report Formatter (ERF)"*
Using DECevent to report system events	*Section 6.5, "Using DECevent"*
Setting up, maintaining, and printing the operator log file	*Section 6.7, "Setting Up, Maintaining, and Printing the Operator Log File"*
Using security auditing	*Section 6.8, "Using Security Auditing"*
Using the Monitor utility to monitor system performance	*Section 6.9, "Monitoring Operating System Performance"*

This chapter explains the following concepts:

Concept	Section
System log files	*Section 6.1, "Understanding System Log Files"*
Error logging	*Section 6.2, "Understanding Error Logging"*
Error Log utility (ERROR LOG)	*Section 6.4.1, "Understanding the Error Log Report Formatter (ERF)"*
DECevent Event Management utility	*Section 6.5.1, "Understanding DECevent"*

Concept	Section
Operator log file	*Section 6.7.1, "Understanding the Operator Log File"*
OPCOM messages	*Section 6.7.2, "Understanding OPCOM Messages"*
Security auditing	*Section 6.8.1, "Understanding Security Auditing"*
Monitor utility (MONITOR)	*Section 6.9.1, "Understanding MONITOR"*

6.1. Understanding System Log Files

In maintaining your system, collect and review information about system events. The operating system provides several log files that record information about the use of system resources, error conditions, and other system events. *Table 6.1, "System Log Files"* briefly describes each file and provides references to sections that discuss the files in more detail.

Table 6.1. System Log Files

Log File	Description	For More Information
Error log file	The system automatically records device and CPU error messages in this file.	*Section 6.2, "Understanding Error Logging"*
Operator log file	The operator communication manager (OPCOM) records system events in this file.	*VSI OpenVMS System Manager's Manual, Volume 1: Essentials* and *Section 6.7, "Setting Up, Maintaining, and Printing the Operator Log File"*
Accounting file	The accounting file tracks the use of system resources.	*Chapter 7, "Tracking Resource Use"*
Security audit log file	The audit server process preallocates disk space to and writes security-relevant system events to this file.	*Section 6.8, "Using Security Auditing"*

6.2. Understanding Error Logging

The error logging subsystem automatically writes error messages to the latest version of the error log file, SYS$ERRORLOG:ERRLOG.SYS. Error log reports are primarily intended for use by VSI support representatives to identify hardware problems. System managers often find error log reports useful in identifying recurrent system failures that require outside attention.

Parts of the Error Logging Subsystem

The error logging facility consists of the parts shown in *Table 6.2, "Parts of the Error Logging Facility"*.

Table 6.2. Parts of the Error Logging Facility

Part	Description
Executive routines	Detect errors and events, and write relevant information into error log buffers in memory.
Error Formatter (ERRFMT)	The ERRFMT process, which starts when the system is booted, periodically empties error log buffers, transforms the descriptions of errors into standard formats, and stores formatted information in an error log file on the system disk. (See *Section 6.3.2, "Maintaining Error Log Files"*.) The Error Formatter allows you to send mail to the SYSTEM account or another user if the ERRFMT process encounters a fatal error and deletes itself. (See *Section 6.3.3, "Using ERRFMT to Send Mail"*.)
Error Log utility (ERROR LOG)	Selectively reports the contents of an error log file. This utility is most useful with error logs written on systems running OpenVMS prior to Version 7.2. You invoke ERF by entering the DCL command ANALYZE/ERROR_LOG. (See *Section 6.4, "Using the Error Log Report Formatter (ERF)"*.)
DECevent	Selectively reports the contents of an event log file. This utility is most useful with error logs written on systems running OpenVMS Versions 7.2 to 7.3. DECevent Version 2.9 and higher includes the Binary Error Log Translation utility. You invoke DECevent by entering the DCL command DIAGNOSE. (See *Section 6.5, "Using DECevent"*.)
Error Log Viewer (ELV)	Selectively reports the contents of an error log file. This utility is most useful with error logs written on systems running OpenVMS Version 7.3 and later. You invoke ELV by entering the DCL command ANALYZE/ERROR_LOG/ELV. (See *Section 6.6, "Using the Error Log Viewer (ELV)"*.)

The executive routines and the Error Formatter (ERRFMT) process operate continuously without user intervention. The routines fill the error log buffers in memory with raw data on every detected error and event. When one of the available buffers becomes full, or when a time allotment expires, ERRFMT automatically writes the buffers to SYS$ERRORLOG:ERRLOG.SYS.

Sometimes a burst of errors can cause the buffer to fill up before ERRFMT can empty them. You can detect this condition by noting a skip in the error sequence number of the records reported in the error log reports. As soon as ERRFMT frees the buffer space, the executive routines resume preserving error information in the buffers.

The ERRFMT process displays an error message on the system console terminal and stops itself if it encounters excessive errors while writing the error log file. *Section 6.3.1, "Restarting the ERRFMT Process"* explains how to restart the ERRFMT process.

6.3. Using the Error Formatter

The Error Formatter (ERRFMT) process is started automatically at boot time. The following sections explain how to perform these tasks:

Task	Section
Restart the ERRFMT process, if necessary	*Section 6.3.1, "Restarting the ERRFMT Process"*
Maintain error log files	*Section 6.3.2, "Maintaining Error Log Files"*
Send mail if the ERRFMT process is deleted	*Section 6.3.3, "Using ERRFMT to Send Mail"*

6.3.1. Restarting the ERRFMT Process

To restart the ERRFMT process, follow these steps:

1. Log in to the system manager's account so that you have the required privileges to perform the operation.

2. Execute the site-independent startup command procedure (STARTUP.COM), specifying ERRFMT as the command parameter, as follows:

```
$ @SYS$SYSTEM:STARTUP ERRFMT
```

Note

If disk quotas are enabled on the system disk, ERRFMT starts only if UIC [1,4] has sufficient quotas.

6.3.2. Maintaining Error Log Files

Because the error log file, SYS$ERRORLOG:ERRLOG.SYS, is a shared file, ERRFMT can write new error log entries while the Error Log utility reads and reports on other entries in the same file.

ERRLOG.SYS increases in size and remains on the system disk until you explicitly rename or delete it. Therefore, devise a plan for regular maintenance of the error log file. One method is to rename ERRLOG.SYS on a daily basis. If you do this, the system creates a new error log file. You might, for example, rename the current copy of ERRLOG.SYS to ERRLOG.OLD every morning at 9:00. To free space on the system disk, you can then back up the renamed version of the error log file on a different volume and delete the file from the system disk.

Another method is to keep the error log file on a disk other than the system disk by defining the logical name SYS$ERRORLOG to be the device and directory where you want to keep error log files; for example:

```
$ DEFINE/SYSTEM/EXECUTIVE SYS$ERRORLOG DUA2:[ERRORLOG]
```

To define this logical name each time you start up the system, add the logical name definition to your SYLOGICALS.COM procedure. See *VSI OpenVMS System Manager's Manual, Volume 1: Essentials* for details.

Be careful not to delete error log files inadvertently. You might also want to adopt a file-naming convention that includes a beginning or ending date for the data in the file name.

6.3.3. Using ERRFMT to Send Mail

The Error Formatter (ERRFMT) allows users to send mail to the system manager or to another designated user if the ERRFMT process encounters a fatal error and deletes itself.

Two system logical names, ERRFMT$_SEND_MAIL and ERRFMT$_SEND_TO, control this feature:

- ERRFMT$_SEND_MAIL

  To enable sending mail, must translate to the string TRUE, and is case insensitive. Any other value disables the sending of mail.

- ERRFMT$_SEND_TO

  Must translate to a user name (the current default is SYSTEM).

  VSI recommends that you do not use distribution lists and multiple user names.

You can define these logical names in one of two ways:

- Dynamically, using DCL DEFINE/SYSTEM commands

  After you make the changes, you must stop and restart ERRFMT for the changes to take effect.

- Permanently, in SYS$STARTUP:SYLOGICAL.COM

  The logical names you define take effect the next time the system is rebooted. The following instructions use this method.

6.3.3.1. Enabling and Disabling ERRFMT to Send Mail

If ERRFMT$_SEND_MAIL is defined to be TRUE, you receive a mail message with a subject line saying that ERRFMT is about to delete itself. The operator log file and the output displayed at the system console, OPA0:, contain more detailed information about the failure encountered and instructions on how to restart ERRFMT; however, you are often not at the console to see this information.

If you are using ERRFMT in one mode, for example, with sending mail enabled, and you want to disable sending mail, use the system manager's account to edit SYS$STARTUP:SYLOGICAL.COM, adding the following command:

```
$ DEFINE/SYSTEM ERRFMT$_SEND_MAIL FALSE
```

To reenable sending mail, use the system manager's account to edit SYS $STARTUP:SYLOGICAL.COM, adding the following command:

```
$ DEFINE/SYSTEM ERRFMT$_SEND_MAIL TRUE
```

6.3.3.2. Sending Mail to Another User

Sending mail to the SYSTEM account is enabled by default. However, you can define ERRFMT $_SEND_TO to send mail to another user if ERRFMT is about to delete itself.

To change the user name to receive mail, use the system manager's account to edit SYS $STARTUP:SYLOGICAL.COM, adding an appropriate logical name DEFINE command. For example:

```
$ DEFINE/SYSTEM ERRFMT$_SEND_TO R_SMITH
```

VSI recommends that you do not use distribution lists and multiple user names.

6.4. Using the Error Log Report Formatter (ERF)

Use the Error Log Report Formatter (ERF) to report selectively on the contents of an error log file. You must have the SYSPRV privilege to run ERF.

Starting with OpenVMS Version 7.2, before using ERF, you must convert error log files using the Error Log Viewer (ELV) or the Binary Error Log Translation utility, which is part of DECevent. For more information about ELV, refer to *Section 6.6, "Using the Error Log Viewer (ELV)"* and to the *VSI OpenVMS System Management Utilities Reference Manual*. For more information about DECevent, refer to *Section 6.5, "Using DECevent"* and to its documentation at the following URL:

```
http://tecVSIubs.cxo.cpqcorp.net/doc_event.html
```

6.4.1. Understanding the Error Log Report Formatter (ERF)

The Error Log Report Formatter (ERF) supports most OpenVMS supported hardware, such as adapters, disks, tapes, CPUs, and memories, but not all communications devices. Some synchronous communications devices are supported.

The operating system automatically writes messages to the latest version of an error log file, SYS $ERRORLOG:ERRLOG.SYS, as the events shown in *Table 6.3, "Types of Events Reported in the Error Log File"* occur.

Table 6.3. Types of Events Reported in the Error Log File

Event	Description
Errors	Device errors, device timeouts, machine checks, bus errors, memory errors (hard or soft error correcting code [ECC] errors), asynchronous write errors, and undefined interrupts
Volume changes	Volume mounts and dismounts
System events	System startups, messages from the Send Message to Error Logger ($SNDERR) system service, and time stamps

You can use ERF to process error log entries for the following forms of optional output:

- Full report of selected entries, which is the default
- Brief report of selected entries
- Summary report of selected entries
- Register dump report of selected device entries
- Binary copy of selected entries
- Binary copy of rejected entries

Section 6.4.2, "Producing Error Log Reports" explains how to produce error log reports. Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for examples of error log reports.

The error reports that ERF produces are useful in two ways:

- They aid preventive maintenance by identifying areas within the system that show potential for failure.
- They aid the diagnosis of a failure by documenting the errors and events that led up to it.

The detailed contents of the reports are most meaningful to VSI support representatives. However, you can use the reports as an important indicator of the system's reliability. For example, using the DCL command SHOW ERROR, you might see that a particular device is producing a relatively high number of errors. You can then use ERF to obtain a more detailed report and decide whether to consult your support representative.

If a system component fails, a VSI support representative can study the error reports of the system activity leading up to and including the failure. If a device fails, you can generate error reports immediately after the failure; for example:

- One report might describe in detail all errors associated with the device that occurred within the last 24 hours.
- Another report might summarize all types of errors for all devices that occurred within the same time period.
- The summary report can put the device errors into a systemwide context.

Your support representative can then run the appropriate diagnostic program for a thorough analysis of the failed device. Using the combined error logging and diagnostic information, your support representative can proceed to correct the device.

Error reports allow you to anticipate potential failures. Effective use of the Error Log Report Formatter in conjunction with diagnostic programs can significantly reduce the amount of system downtime.

6.4.2. Producing Error Log Reports

You enter the DCL command in the following format:

```
ANALYZE/ERROR_LOG /qualifier(s) filespec,...
```

where:

`qualifier`	Specifies the function the ANALYZE/ERROR_LOG command is to perform.
`filespec`	Specifies one or more files that contain information to be interpreted for the error log report.

Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for details about the command and its parameters and for examples of error log reports.

ERF issues error messages for inconsistent ERF entries. Use the Help Message facility to look up explanations and suggested user actions for these messages.

6.4.3. Producing a Full Error Log Report

The following steps show how to produce an error log report for all entries in the error log file and how to print the report:

1. Either log in to the SYSTEM account or ensure that you have the SYSPRV privilege. (You must have privilege to access the error log file.) For example:

   ```
   $ SET PROCESS/PRIVILEGE=SYSPRV
   ```

2. Set your default disk and directory to SYS$ERRORLOG:

   ```
   $ SET DEFAULT SYS$ERRORLOG
   ```

3. Examine the error log directory to see which error log file you want to analyze:

   ```
   $ DIRECTORY
   ```

4. To obtain a full report of the current error log file, enter the following command:

   ```
   $ ANALYZE/ERROR_LOG/OUTPUT=ERRORS.LIS
   ```

5. Print a copy of the report, using the file name you specified with the /OUTPUT qualifier:

   ```
   $ PRINT ERRORS.LIS
   ```

Example

```
$ SET PROCESS/PRIVILEGE=SYSPRV
$ SET DEFAULT SYS$ERRORLOG
$ DIRECTORY  ❶
Directory SYS$SYSROOT:[SYSERR]

ERRLOG.OLD;2  ERRLOG.OLD;1  ERRLOG.SYS;1

Total of 3 files.
$ ANALYZE/ERROR_LOG/OUTPUT=ERRORS.LIS ERRLOG.OLD ❷
$ PRINT ERRORS.LIS ❸
```

The following list explains the commands in the example.

❶ The DIRECTORY command lists all the files in the SYS$ERRORLOG directory. The directory contains three files: two old error log files and the current error log file, ERRLOG.SYS.

❷ The ANALYZE/ERROR_LOG command writes a full report to a file called ERRORS.LIS, using the most recent ERRLOG.OLD file as input.

❸ The PRINT command prints ERRORS.LIS.

6.4.4. Using Other ERF Report Options

This section briefly explains how to specify report formats and produce a report of selected entries.

Table 6.4, "ERF Report Options" contains error log report options. For more details about options and examples of error log reports using options, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

Table 6.4. ERF Report Options

In Order To...	You Can...
Specify report formats	Change report formats by using qualifiers, including the following ones: • /BINARY – to convert binary error log records to ASCII text or to copy error log records to a specified output file.

In Order To...	You Can...
	• /BRIEF – to create a brief report. • /REGISTER_DUMP – to generate, in a hexadecimal longword format, a report that consists of device register information (used in conjunction with the /INCLUDE qualifier). • /REJECTED – to specify the name of a file that will contain binary records for rejected entries.
Specify a display device for reports	Use the /OUTPUT qualifier to send reports to a terminal for display or to a disk or magnetic tape file. By default, the system sends the report to the SYS$OUTPUT device. Because error log reports are 72 columns wide, you can display them on the terminal screen.
Produce a report of selected entries	Use qualifiers to produce error log reports for specific types of events and for a specified time interval. For example, you can process error log entries by selecting a time interval using the /SINCE, /BEFORE, or /ENTRY qualifiers. You can specify error log entries for specific events by using the qualifiers /INCLUDE and /EXCLUDE. These qualifiers form a filter to determine which error log entries are selected or rejected. In addition, you can generate error log reports for one or more OpenVMS Cluster members by using the /NODE qualifier.
Exclude unknown error log entries	By default, when ERF encounters an unknown device, CPU, or error log entry, the utility produces the entry in hexadecimal longword format. Exclude these entries from the report by specifying /EXCLUDE=UNKNOWN_ENTRIES in the command line.

6.5. Using DECevent

The DECevent Event Management utility (DECevent) provides an interface between a system user and the operating system's event log files.

Note

On Alpha DS, ES, and GS systems (other than the AlphaServer GS60 and GS140 systems) running OpenVMS, use the Error Log Viewer (ELV) or Web-Based Enterprise Services (WEBES). WEBES includes the VSI Analyze, VSI Crash Analysis Tool, and the Revision and Configuration Management (RCM) tools. You can find WEBES and its documentation on the VSI System Tools CD-ROM, which is included in the OpenVMS Version 7.3-2 CD-ROM package.

You cannot use WEBES on the AlphaServer GS60 or the AlphaServer GS140. DECevent and the WEBES tools can be used together in a cluster.

6.5.1. Understanding DECevent

DECevent allows system users to produce ASCII reports derived from system event entries. The format of the ASCII reports depends on the command entered on the command language interpreter (CLI) with a maximum character limit of 255 characters.

DECevent uses the error log file, SYS$ERRORLOG:ERRLOG.SYS, as the default input file, unless you specify another input file.

Event reports are useful for determining preventive maintenance by helping to identify areas within the system showing potential failure. Event reports also aid in the diagnosis of a failure by documenting events that led to the failure.

The contents of the event reports are most meaningful to your VSI support representative. However, you can use the event reports as an indicator of system reliability. For example, while using the DCL command SHOW ERROR, you might see that a particular device is producing a higher than normal number of events. You can use DECevent to obtain various detailed reports and determine if you need to contact your VSI support representative.

If a system component fails, your VSI support representative can use the event reports to create a history of events leading up to and including the failure.

Used in conjunction with diagnostic programs, event reports significantly reduce the amount of system down time.

DECevent Report Types

DECevent produces five types of reports:

Report Type	Description
Full (default)	Provides a translation of all available information for each entry in the event log.
Brief	Provides a translation of key information for each entry in the event log.
Terse	Provides binary event information and displays register values and other ASCII messages in a condensed format.
Summary	Provides a statistical summary of the event entries in the event log.
Fast Error (FSTERR)	Provides a quick, one-line per-entry report of your event log for a variety of disk devices.

These report types are mutually exclusive; in other words, you can select only one report type in a command.

Section 6.5.5, "Producing DECevent Reports" contains examples of types of reports. The *VSI OpenVMS System Management Utilities Reference Manual* contains additional examples of the types of reports produced by DECevent.

The following sections explain how to use DECevent:

Task	Section
Invoking and exiting DECevent	*Section 6.5.2, "Invoking and Exiting DECevent"*
Using DECevent qualifiers	*Section 6.5.3, "Using DECevent Qualifiers"*
Using additional DECevent commands	*Section 6.5.4, "Using Additional*

Task	Section
	DECevent Commands"
Producing DECevent reports	*Section 6.5.5, "Producing DECevent Reports"*

In addition, restrictions are listed in *Section 6.5.6, "DECevent Restrictions"*.

6.5.2. Invoking and Exiting DECevent

To invoke DECevent, enter the DCL command DIAGNOSE using the following syntax:

```
DIAGNOSE [/primary qualifier][/secondary qualifier[,...][file-spec][,..]
```

Note that you do not need to enter the /TRANSLATE qualifier on the command line because it is the default primary qualifier.

A brief discussion of valid qualifiers, their uses, and their order is provided in the following sections. For a more detailed discussion, see the *DECamds User's Guide* This guide is available online at the following web site:

```
http://techpubs.cxo.cpqcorp.net/doc_event.html
```

The *DECevent User's Guide* is available in several formats.

To exit DECevent, press Ctrl/C and the Return key (to display the system prompt).

You must have SYSPRV privilege to run DECevent; however, only read access is required to access the ERRLOG.SYS file. You must have the DIAGNOSE privilege for the /CONTINUOUS primary qualifier to work, enabling the continuous display of events on a terminal screen.

6.5.3. Using DECevent Qualifiers

The DECevent qualifiers shown and described in *Table 6.5, "Primary Qualifiers"* and *Table 6.6, "(Optional) Secondary Qualifiers"* allow you to change the format of the reports that DECevent produces.

Table 6.5. Primary Qualifiers

Qualifier	Description
/ANALYZE	Provides analysis of the event log or real-time analysis of the event logging utilities.
/BINARY	Controls whether the binary error log records are converted to ASCII text or copied to the specified output file. Do not use this qualifier with any report type qualifier (/FULL, /BRIEF, /TERSE, /SUMMARY, and /FSTERR) or with the /OUTPUT qualifier.
/CONTINUOUS	Specifies that events are formatted in real time, as they are logged by the operating system event logger.
/DUMP	Specifies the output to be a brief report followed by a dump of information from the input event log file.
/INTERACTIVE	Allows users to exit from the command line interface and enter the DECevent interactive command shell.

Qualifier	Description
/TRANSLATE (default)	Provides translation of event log files into reports.

Table 6.6. (Optional) Secondary Qualifiers

Qualifier	Description
/BEFORE	Specifies that only those entries dated earlier than the stated date and time are to be selected for the event report.
/BRIEF	Generates a brief report
/ENTRY	Generates a report that includes the specified entry range or starts at the specified entry number.
/EXCLUDE	Excludes events generated by the specified device class, device name, or error log entry type from the report.
/FSTERR	Generates a quick, one-line-per-entry report of an event log entry for disks and tapes.
/FULL (default)	Generates a full report, which provides all available information about an event log entry.
/INCLUDE	Includes events generated by the specified device class, device name, or error log entry type in the report.
/LOG	Controls whether informational messages that specify the number of entries selected and rejected for each input file are sent to SYS$OUTPUT.
/NODE	Generates a report consisting of event entries for specific nodes in an OpenVMS Cluster system.
/OUTPUT	Specifies the output file for the report.
/REJECTED	Allows you to specify the name of a file that will contain binary records for rejected entries.
/SINCE	Specifies that only those entries dated later than the stated date and time are to be selected for the report.
/SUMMARY	Generates an event report that consists of a statistical summary.
/TERSE	Generates an event report consisting of binary event information, register values, and ASCII messages in a condensed format.

Do not use the /BINARY qualifier with any report type qualifier (/FULL, /BRIEF, /TERSE, /SUMMARY, and /FSTERR) or with the /OUTPUT qualifier.

Privileges Required

- You must have SYSPRV privilege to run DECevent; however, only read access is required to access the ERRLOG.SYS file.
- You must have the DIAGNOSE privilege for the /CONTINUOUS qualifier to work, enabling the continuous display of events on a terminal screen.

6.5.4. Using Additional DECevent Commands

In addition to the qualifiers listed in *Table 6.5, "Primary Qualifiers"* and *Table 6.6, "(Optional) Secondary Qualifiers"*, DECevent contains a set of DIRECTORY commands and a set of SHOW commands:

- DIRECTORY commands

  These commands allow you to display a list of rule sets that DECevent needs to translate events into a readable format. (A **rule set** is a software routine or function that is analogous to an executable file.)

  The following DIRECTORY commands are currently implemented in DECevent:

  - DIRECTORY EVENT

    This command lists all rule sets associated with event translation.

  - DIRECTORY CANONICAL

    This command lists all rule sets associated with event reports.

- SHOW commands

  These commands allow a user to view specific settings and selections. The following SHOW commands are currently implemented in DECevent:

  - SHOW SELECT

    By appending a specific selection keyword name to the SHOW SELECT command, you view only that selection keyword.

  - SHOW SETTINGS

    By appending a specific setting's name to the SHOW SETTINGS command, you view only that setting's name and value.

6.5.5. Producing DECevent Reports

This section contains examples of DECevent commands and reports.

6.5.5.1. Producing a Full Report

To produce a full report, use the /FULL qualifier. The full report format provides a translation of all available information for each entry in the event log. The full report is the default report type if a report type is not specified in the command line.

Both of the following commands will produce a full report format:

```
$ DIAGNOSE/FULL
$ DIAGNOSE
```

(/FULL is the default.)

Example 6.1, "Full Report Format" shows the format of a full report.

Example 6.1. Full Report Format

```
************************ ENTRY    1 ************************
```

```
Logging OS                          1. OpenVMS
System Architecture                 2. Alpha
OS version                             V7.3
Event sequence number            1583.
Timestamp of occurrence                18-APR-2016 09:21:18
System uptime in seconds        58004.
Error mask                  x00000000
Flags                           x0001  Dynamic Device Recognition present
Host name                              COGENT

Alpha HW model                         DEC 3000 Model 400
System type register        x00000004  DEC 3000
Unique CPU ID               x00000002
mpnum                       x000000FF
mperr                       x000000FF

Event validity                     -1. Unknown validity code
Event severity                     -1. Unknown severity code
Entry type                        100.
Major Event class                   3. IO Subsystem

IO Minor Class                      1. MSCP
IO Minor Sub Class                  5. Logged Message

---- Device Profile ----
Vendor
Product Name                           RAID 0 - Host Based
Unit Name                              COGENT$DPA
Unit Number                       10.
Device Class                    x0001  Disk

---- IO SW Profile ----
VMS DC$_CLASS                       1.
VMS DT$_TYPE                      175.

---- MSCP Logged Msg ----

Logged Message Type Code           22. RAID Message
RAID Event Type                     8. Remove Member
Distinguished Member                0.
Member Index                        1.
RAID Urgency                        4. Global Disk Error
RAID Status                 x00180009  Bit 00 - Reduced
                                       Bit 03 - Striped
                                       Bit 19 - FE Dis FE
                                       Bit 20 - BC Buff Copy Off
RAIDset Name                           KGB
************************************************************
```

6.5.5.2. Producing a Brief Report

To produce a brief report, use the /BRIEF qualifier. The brief report format provides translation of key information for each entry in the event log. For example:

```
$ DIAGNOSE/BRIEF
```

Example 6.2, "Brief Report Format " shows the format of a brief report.

Example 6.2. Brief Report Format

```
************************ ENTRY    1 ************************



Logging OS                          1. OpenVMS
System Architecture                 2. Alpha
OS version                             V7.3
Event sequence number            1583.
Timestamp of occurrence                18-APR-2016 09:21:18
System uptime in seconds        58004.
Error mask                   x00000000
Host name                              COGENT

Alpha HW model                         DEC 3000 Model 400
System type register         x00000004  DEC 3000
Unique CPU ID                x00000002
mpnum                        x000000FF
mperr                        x000000FF

Event validity                     -1. Unknown validity code
Event severity                     -1. Unknown severity code
Major Event class                   3. IO Subsystem

IO Minor Class                      1. MSCP
IO Minor Sub Class                  5. Logged Message

---- Device Profile ----
Vendor
Product Name                           RAID 0 - Host Based
Unit Name                              COGENT$DPA
Unit Number                       10.
Device Class                   x0001  Disk



Logged Message Type Code           22. RAID Message
RAID Event Type                     8. Remove Member
Distinguished Member                0.
Member Index                        1.
RAID Urgency                        4. Global Disk Error
RAID Status                  x00180009  Bit 00 - Reduced
                                        Bit 03 - Striped
                                        Bit 19 - FE Dis FE
                                        Bit 20 - BC Buff Copy Off
RAIDset Name                           KGB
*************************************************************
```

6.5.5.3. Producing a Terse Report

To produce a terse report, use the /TERSE qualifier. The terse report format provides binary event information and displays register values and other ASCII messages in a condensed format. For example:

```
$ DIAGNOSE/TERSE
```

Example 6.3, "Terse Report Format " shows the format of a terse report.

Example 6.3. Terse Report Format

```
************************ ENTRY    1 ************************


Logging OS                            1.
System Architecture                   2.
OS version                    V7.3
Event sequence number               1583.
Timestamp of occurrence       2000041809211800
System uptime in seconds           58004.
Error mask                     x00000000
Flags                              x0001
Host name                     COGENT

Alpha HW model                DEC 3000 Model 400
System type register           x00000004
Unique CPU ID                  x00000002
mpnum                          x000000FF
mperr                          x000000FF

Event validity                       -1.
Event severity                       -1.
Entry type                          100.
Major Event class                     3.

IO Minor Class                        1.
IO Minor Sub Class                    5.

---- Device Profile ----
Vendor
Product Name                  RAID 0 - Host Based
Unit Name                     COGENT$DPA
Unit Number                          10.
Device Class                       x0001

---- IO SW Profile ----
VMS DC$_CLASS                         1.
VMS DT$_TYPE                        175.

---- MSCP Logged Msg ----

Logged Message Type Code             22.
RAID Event Type                       8.
Distinguished Member                  0.
Member Index                          1.
RAID Urgency                          4.
RAID Status                    x00180009
RAIDset Name                  KGB

**************************************************
```

6.5.5.4. Producing a Summary Report

To produce a summary report, use the /SUMMARY qualifier. The summary report format provides a statistical summary of the event entries in the event log. For example:

```
$ DIAGNOSE/SUMMARY
```

Example 6.4, "Summary Report Format " shows the format of a summary report.

Example 6.4. Summary Report Format

```
 SUMMARY OF ALL ENTRIES LOGGED ON NODE COGENT

     IO Subsystem
       MSCP                                  9.
       Host Based RAID                       3.

 DATE OF EARLIEST ENTRY                18-APR-2016 09:21:18
 DATE OF LATEST ENTRY                  12-MAY-2016 10:44:54
```

6.5.5.5. Producing a Fast Error (FSTERR) Report

To produce a Fast Error report, use the /FSTERR qualifier. For example:

```
$ DIAGNOSE/FSTERR
```

The Fast Error report provides a quick, one-line-per-entry report of your event log for a variety of disk devices. This makes event analysis and system troubleshooting much easier by eliminating extraneous event information. For example:

```
$ DIAGNOSE/FSTERR [infile]
```

A Fast Error report is shown in *Example 6.5, "Fast Error (FSTERR) Report Format "*.

Example 6.5. Fast Error (FSTERR) Report Format

```
 Drive/
                                    MSCP                Physical     HSC
 Volume
  Drive Name  yymmdd hhmmss Entry Evnt LED   LBN    Cyl Hd Sec RA  RP
 Serial
============= ============= ===== ==== === ======= ==== == === === ==
 ======
  LUKE$DUA070 921119 160754     3 00EB         255             70  71
 V00717
  LUKE$DUA070 921119 160754     4 00EB         255             70  71
 V00717
HSC015$DUA028 910323 113204     5 00EB                         70  51
 V15039
HSC015$DUA028 910323 113204     6 00EB                         71  51
 V15039
 BATES$DUA197 921118 002116     7 00EB                         72  32
 V17524
CHEWIE$DUA101 911205 114908     8 00EB                         73  81 V
 17
PMASON$DUA006 921207 165007    15 00EB         255             90  42
 D23387
PMASON$DUA006 921207 165007    16 00EB         255             90  42
 D23387
  C3P0$DUA242 870218 060031    17 01AB                         90  40
 D48575
  CHER$DU2132*901008 231053    18 00EB                         92  81 D
 2345
```

The Fast Error report includes the information needed by a VSI support representative to troubleshoot a problem with a tape or disk device.

6.5.6. DECevent Restrictions

When you use the DECevent utility, note some of the restrictions listed in this section.

Page File Quota

Sometimes, if the page file quota is exceeded, DECevent will terminate and return you to the system prompt. If this happens, invoke the last command.

Logical File Names

DECevent does not translate as input any logical defined as a search list of file names. For example:

```
$ DEFINE EVENT_LOG DISK1:[EVENTS]EVENT_LOG1.SYS,DISK1:EVENT_LOG.SYS
$ DIAGNOSE/ANALYZE EVENT_LOG

DECevent T1.0 FT2
_DIAGNOSE-FAT: Analyze - No files found ' event_log '
_DIAGNOSE-FAT: An error occurred while executing a command ruleset
_DIAGNOSE-INF: No Error Messages to send in thread 1
```

Log File Purging

DECevent does not automatically purge log files. Set the version limit on the files and directory to your preference. For example:

```
$ SET FILE/VERSION=3 DIA_ACTIVITY.LOG
```

System-Initiated Call Logging

When a system running DECevent is shut down and rebooted, DECEVENT$STARTUP.COM does not define FMGPROFILE logicals. This can interfere with proper logging of system initiated call logging (SICL) due to missing customer profile information in the SICL message text.

Unrecognized Messages

The DIAGNOSE command does not recognize error log messages logged using the $SNDERR system service.

6.6. Using the Error Log Viewer (ELV)

The Error Log Viewer (ELV), like ERF and DECevent, selectively reports the contents of an error log file. ELV is most useful with error logs written on systems running OpenVMS Version 7.3-1 and later, while DECevent and ERF are most useful with earlier versions of OpenVMS.

For more detailed information about ELV, refer to the ELV chapter in *VSI OpenVMS System Management Utilities Reference Manual: A-L*.

6.6.1. Understanding the Error Log Viewer (ELV)

The Error Log Viewer (ELV) utility allows you to quickly examine, from the command line, an error log file in a user-readable format. You can do this prior to making a decision that the data warrants a more comprehensive analysis with a tool such as the System Event Analyzer (SEA).

ELV provides detailed information for all events belonging to the event types shown in *Table 6.7, "Types of Events That ELV Fully Supports"*. Event types are grouped by the event classes shown in the same table.

Table 6.7. Types of Events That ELV Fully Supports

Event Class	Event Types
Control entries	System startups, time stamps, operator and network messages, indictment events, ERRLOG.SYS created messages, and messages from the Send Message to Error Logger ($SNDERR) system service
Volume changes	Volume mounts and dismounts
Bugchecks	System bugchecks, user bugchecks, and crash restarts
Machine checks	Correctable error throttling notifications; 6A0/6B0 recoverable uncorrectable errors
Device errors	Software parameters

ELV provides detailed information for some events belonging to the event types shown in *Table 6.8, "Types of Events That ELV Fully Supports"*. Events types are grouped by the event classes shown in the same table.

Table 6.8. Types of Events That ELV Fully Supports

Event Class	Event Types
Machine checks	620 system correctable errors, 630 processor correctable errors, 660 system uncorrectable errors, 670 processor uncorrectable errors, 680 system events, console data logs
Device errors	Device errors, device timeouts, asynchronous device attentions
Unsolicited MSCP	Logged MSCP messages

6.6.2. Invoking ELV

To invoke ELV, enter the following DCL command:

```
$ ANALYZE/ERROR_LOG/ELV
```

If you do not enter an ELV command, the utility enters interactive shell mode and displays the ELV prompt:

```
ELV>
```

You can then enter an ELV command. After ELV executes the command, it again displays the ELV> prompt.

To return directly to DCL after executing an ELV command from the ELV prompt, use the /NOINTERACTIVE qualifier.

You can also enter an ELV command directly from DCL; for example:

```
$ ANALYZE/ERROR_LOG/ELV TRANSLATE ERRLOG.SYS;42
```

After ELV executes the command, you are returned to the DCL prompt by default.

To enter interactive shell mode after executing an ELV command directly from DCL, use the /INTERACTIVE qualifier.

6.6.3. Principal ELV Commands

The commands shown in *Table 6.9, "Principal ELV Commands"* represent the principal ELV operations.

Table 6.9. Principal ELV Commands

Command	Description
CONVERT	Converts and writes events from one or more binary error log files written in the newer format to a single new error log file written in the older format. The new file can then be read by ANALYZE/ERROR_LOG. This command is primarily used to enable translation of older error log events whose translation is not supported by ELV.
DUMP	Writes events from one or more binary error log files to a single new ASCII output file in an OpenVMS dump-style format.
TRANSLATE	Performs a bit-to-text translation of events from one or more binary error log files and writes the resulting reports to the terminal or to a single new ASCII output file.
WRITE	Performs an image copy of events from one or more binary error log files to a single new binary error log file.

Using various qualifiers that are common to all of these commands, you can select or reject the events to be processed by one of these commands. For example, if you specify TRANSLATE /SINCE=YESTERDAY, you translate all valid events that have occurred since yesterday.

6.6.4. Standard Reports Using the TRANSLATE Command

You can use the TRANSLATE command to produce standard reports of various detail levels. This is the primary function of the ELV utility.

To specify the detail level of a standard report, you can use the /BRIEF, /FULL, or /ONE_LINE qualifier with the TRANSLATE command, as outlined in *Table 6.10, "Standard Report Detail Levels"*, or you can accept the default report by omitting a detail level qualifier. In addition to these qualifiers, you can use the /TERSE qualifier to obtain a standard report that contains less interpretation of the data, regardless of detail level.

Table 6.10. Standard Report Detail Levels

Detail Level	Qualifier	Description
One-line	/ONE_LINE	The header information is the only information that is included in the standard report.
Brief	/BRIEF	Only the most essential information is included with the header information.
Default	(None)	Only the most commonly useful event information is included with the header information.

Detail Level	Qualifier	Description
Full	/FULL	All event information is included with the header information.

6.6.4.1. Example of a Standard Report

Example 6.6, "Standard and Summary Reports" shows a standard report followed by a summary report. To produce only the standard report (and omit the summary report, which is included by default), use the /NOSUMMARY qualifier. To produce only the summary report (and omit the standard report, which is included by default), use the /SUMMARY qualifier.

Example 6.6. Standard and Summary Reports

```
Output for SYS$COMMON:[SYSEXE.ERRLOGS]EXAMPLE.DAT;1
EVENT EVENT_TYPE_____________________________ TIMESTAMP_________________ NODE__
 EVENT_CLASS_____________________
1 Volume Mount 14-AUG-2003 13:31:39.12 FRANZ VOLUME_CHANGES
DESCRIPTION_____________________________ RANGE___ VALUE_______________
 TRANSLATED_VALUE_________________
Hardware Architecture                              4                  Alpha
Hardware System Type                               35                 Wildfire
Logging CPU                                        3
Number of CPU's in Active Set                      4
System Marketing Model                             1968               COMPAQ AlphaServer
 GS160
Error Mask                                 <31:00>: 0x00000003
Seconds Since Boot                                 17
Error Sequence Number                              46
DSR String                                         AlphaServer GS160 6/731
Operating System Version                           X9WY-SSB

Owner UIC of the Volume                            65537
Unit Operation Count                               378
Device Unit Number                                 200
Device Generic Name                                FRANZ$DKB
Volume Number within Set                           0
Number of Volumes within Set                       0
Volume Label                                       OPAL_X9WY

ERROR_LOG_SUMMARY_____________________________________________________

Total number of events:                            1
Number of the first event:                         1
Number of the last event:                          1
Earliest event occurred:                           14-AUG-2003 13:31:39.12
Latest event occurred:                             14-AUG-2003 13:31:39.12
Number of events by event class:
VOLUME_CHANGES                                     1
```

6.7. Setting Up, Maintaining, and Printing the Operator Log File

The following sections describe the contents of the operator log file and OPCOM messages. They also explain how to perform the following tasks, which require OPER privilege:

Task	Section
Setting up the operator log file	*Section 6.7.3, "Setting Up the Operator Log File"*

Task	Section
Maintaining the operator log file	*Section 6.7.4, "Maintaining the Operator Log File"*
Printing the operator log file	*Section 6.7.5, "Printing the Operator Log File"*

6.7.1. Understanding the Operator Log File

The operator log file (SYS$MANAGER:OPERATOR.LOG) records system events and user requests that the operator communication manager (OPCOM) sends to the operator terminal. This recording occurs even if all operator terminals have been disabled. By default, OPCOM starts when you boot your system. (For more information about OPCOM, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.)

You can use the operator log file to anticipate and prevent hardware and software failures and to monitor user requests for disk and magnetic tape operations. By regularly examining the operator log file, you can often detect potential problems and take corrective action.

The size of and access to the OPERATOR.LOG file (or to the file pointed to by the logical OPC $LOGFILE_NAME) is limited by the size and access of the disk device on which it resides. If disk device does not have enough room to write to the log file or if access to the device in any other way is restricted, records might be missing from the log file.

6.7.2. Understanding OPCOM Messages

The following sections describe the types of messages that appear in the operator log file.

Type of Message	Section
Initialization messages	*Section 6.7.2.1, "Initialization Messages"*
Device status messages	*Section 6.7.2.2, "Device Status Messages"*
Terminal enable and disable messages	*Section 6.7.2.3, "Terminal Enable and Disable Messages"*
User request and operator reply messages	*Section 6.7.2.4, "User Request and Operator Reply Messages"*
Volume mount and dismount messages	*Section 6.7.2.5, "Volume Mount and Dismount Messages"*

Type of Message	Section
System parameter messages	*Section 6.7.2.6, "System Parameter Messages"*
Security alarm messages	*Section 6.7.2.7, "Security Alarm Messages"*

Section 6.7.2.8, "Contents of an Operator Log File" contains an example of typical kinds of messages found in an operator log file.

6.7.2.1. Initialization Messages

When you enter the REPLY/LOG command, the system closes the current operator log file and creates and opens a new version of the file. The system records all subsequent OPCOM messages in the new log file.

When you create a new log file, the first message recorded in it is an initialization message. This message shows the terminal name of the operator who initialized the log file, and the log file specification. This message appears in the following format:

```
%%%%%%%%%%% OPCOM, <dd-mmm-yyyy hh:mm:ss.cc> %%%%%%%%%%%
Logfile has been initialized by operator
Logfile is <logfile-specification>
```

Example

```
%%%%%%%%%%% OPCOM, 19-APR-2002 12:29:24.52 %%%%%%%%%%%
Logfile has been initialized by operator _MARS$VTA2:
Logfile is HOMER::SYS$SYSMOND:[SYSMGT]OPERATOR.LOG;43
```

6.7.2.2. Device Status Messages

Some I/O drivers send messages to OPCOM concerning changes in the status of the devices they control. For example, when a line printer goes off line, an OPCOM message appears in the operator log file at periodic intervals until you explicitly return the device to online status.

The device status message appears in the operator log file in the following format:

```
%%%%%%%%%%%  OPCOM <dd-mmm-yyyy hh:mm:ss.cc> %%%%%%%%%%%
Device <device-name> is offline
```

This message can appear for card readers, line printers, and magnetic tapes.

6.7.2.3. Terminal Enable and Disable Messages

The following sections explain commands you can give to enable and disable terminals as operator terminals (or consoles) and explanations of the corresponding messages that appear in the operator log file.

REPLY/ENABLE Messages

To designate a terminal as an operator terminal, enter the REPLY/ENABLE command from the desired terminal. OPCOM confirms the request by displaying messages in the following format at the operator terminal and in the operator log file:

```
%%%%%%%%%%%  %OPCOM dd-mmm-yyyy hh:mm:ss.cc  %%%%%%%%%%%
Operator <terminal-name> has been enabled, username <user-name>

%%%%%%%%%%%  %OPCOM dd-mmm-yyyy hh:mm:ss.cc  %%%%%%%%%%%
Operator status for operator <terminal-name>
<status-report>
```

These messages tell you which terminal has been established as an operator terminal and lists the requests the terminal can receive and respond to.

You can also designate a terminal as an operator terminal for a particular function by entering the REPLY/ENABLE= `class` command.

If you enter the command REPLY/ENABLE=TAPES, for example, OPCOM displays messages similar to the following ones:

```
%%%%%%%%%%%  %OPCOM 19-APR-2016 10:25:35.74 %%%%%%%%%%%
Operator _ROUND$OPA1: has been enabled, username SYSTEM

%%%%%%%%%%%  %OPCOM 19-APR-2016 10:25:38.82 %%%%%%%%%%%
Operator status for operator _ROUND$OPA1:
TAPES
```

OPCOM confirms that the terminal is established as an operator terminal and indicates that the terminal can only receive and respond to requests concerning magnetic-tape-oriented events, such as the mounting and dismounting of tapes.

REPLY/DISABLE Messages

A terminal that you designate as an operator terminal automatically returns to nonoperator status when the operator logs out. To return the terminal to normal (nonoperator) status without logging out, enter the REPLY/DISABLE command from the terminal.

OPCOM confirms that the terminal is no longer an operator terminal by displaying a message both at the operator terminal and in the operator log file. The message, which tells you which terminal has been restored to nonoperator status and when the transition occurred, has the following format:

```
%%%%%%%%%%  %OPCOM <dd-mmm-yyyy hh:mm:ss.cc> %%%%%%%%%%
Operator <terminal-name> has been disabled, username <user-name>
```

If you designate a terminal as an operator terminal and only partial operator status is disabled, OPCOM displays a status message. This message lists which requests the terminal can still receive and respond to. This message is displayed at the operator terminal and in the operator log file in the following format:

```
%%%%%%%%%%  %OPCOM  <dd-mmm-yyyy hh:mm:ss.cc>  %%%%%%%%%%
Operator status for operator <terminal-name>
<status-report>
```

For example, suppose you designate a terminal as an operator terminal that receives messages concerning magnetic tapes and disks, as well as messages intended for the special site-specific operator class known as OPER10. Later, you relinquish the terminal's ability to receive messages concerning tapes. When you enter the REPLY/DISABLE=TAPES command, OPCOM returns a message like the following one:

```
%%%%%%%%%%  %Opcom 19-APR-2016 09:23:45.32  %%%%%%%%%%
Operator status for operator TTA3
DISKS, OPER10
```

This message tells you that terminal TTA3 still receives and can respond to messages about disks and messages directed to OPER10.

6.7.2.4. User Request and Operator Reply Messages

To communicate with the operator, the user enters the REQUEST command, specifying either the /REPLY or /TO qualifier. The following table contains explanations of these qualifiers:

Command	Explanation

REQUEST/REPLY	The request is recorded in the operator log file in the following format:
	`%%%%%%%%%% %OPCOM <dd-mmm-yyyy hh:mm:ss.cc> %%%%%%%%%%`
	`Request <request-id>, from user <user-name> on <node-name>`
	`<_terminal-name:>, <"message-text">`
	This message tells you which user sent the message, the time the message was sent, the request identification number assigned to the message, the originating terminal, and the message itself.

REQUEST/TO	The request is recorded in the operator log file in the format shown in the REQUEST/REPLY example, but without a request identification number:
	`%%%%%%%%%% %OPCOM, <dd-mmm-yyyy hh:mm:ss.cc> %%%%%%%%%%`
	`Request from user <user-name> on <node-name>`
	`<_terminal-name:>, <"message-text">`

Messages also differ depending on how you reply to a user:

Command	Explanation

REPLY/TO	The response is recorded in the operator log file in the following format:
	`response message`
	`<hh:mm:ss.cc>, request <request-id> completed`
	`by operator <terminal-name>`
	This message indicates how the operator responded to the user's request, as well as when the response was entered and which operator responded.

REPLY/ABORT	The response is recorded in the operator log file in the following format:
	`<hh:mm:ss.cc>, request <request-id> was aborted`
	`by operator <terminal-name>`

REPLY/PENDING	The response is not recorded in the operator log file because the request has not yet been completed (that is, the request has not been fulfilled or aborted).

When a user enters a REQUEST/REPLY command and you have disabled all terminals as operators' terminals, OPCOM records all subsequent users' requests in the log file, but returns a message to the user indicating that no operator coverage is available.

All other OPCOM responses to REPLY commands, except responses involving the REPLY/ENABLE, REPLY/DISABLE, and REPLY/LOG commands, are not logged in the operator log file.

6.7.2.5. Volume Mount and Dismount Messages

Perhaps the widest range of operator messages occurs with volume mounts and dismounts; for example:

```
%%%%%%%%%%  OPCOM, 19-APR-2016 22:41:07.54  %%%%%%%%%%
message from user SYSTEM
Volume "KLATU       " dismounted, on physical device MTA0:
15-APR-2016 22:42:14.81, request 2 completed by operator OPA0
```

6.7.2.6. System Parameter Messages

Users with the appropriate privileges can change the following sets of values for system parameters:

Values	Description
Current	Values stored in the default parameter file on disk and used to boot the system
Active	Values stored in memory and used while the system is running

When the system boots, it reads the current values into memory, creating active values. An active value remains equal to the current value until you change either value.

Users can make the following changes to active and current system parameters:

- Active system parameters – Users with CMKRNL privilege can use the System Management utility (SYSMAN) or the System Generation utility (SYSGEN) to change system parameters in the running (active) system. Users can change only those active values that are categorized as *dynamic* system parameters.
- Current system parameters – Users with SYSPRV privilege can use SYSMAN or SYSGEN to change system parameters in the current system.

Note

VSI recommends that you use AUTOGEN or SYSMAN, not SYSGEN, to change system parameters, as explained in *Section 1.2, "Recommended Method for Changing Parameter Values"*.

SYSGEN modifications to the ACTIVE and CURRENT parameters may be monitored and reported by enabling the security auditing subsystem SYSGEN class. For example SET AUDIT/ENABLE=SYSGEN/ALARM or SET AUDIT/ENABLE=SYSGEN/AUDIT. In addition, changes to the CURRENT parameters will also send a message to the operator communication manager (OPCOM) to record the event in the operator log and notify any operator terminals and the operator console. Such a message will have a format similar to this:

```
%OPCOM, 15-APR-2019 16:04:06.30, message from user SYSTEM
%SYSGEN-I-WRITECUR, CURRENT system parameters modified by process ID
 00160030 into file ALPHAVMSSYS.PAR
```

6.7.2.7. Security Alarm Messages

Alarm messages are sent to the security operator terminal when selected events occur. See *Section 6.8.6, "Enabling a Terminal to Receive Alarm Messages"* for instructions about how to enable a terminal to receive security alarm messages.

Example

The following example shows a security alarm OPCOM message after a change to JTQUOTA:

```
%%%%%%%%%%  OPCOM   6-JAN-2016 10:41:21.10  %%%%%%%%%%
Message from user AUDIT$SERVER on BISCO
Security alarm (SECURITY) and security audit (SECURITY) on BISCO, system
 id:
20353
Auditable event:     System UAF record modification
Event time:          6-JAN-2016 10:41:20.69
PID:                 00600123
Process name:        SYSTEM
Username:            SYSTEM
Process owner:       [SYSTEM]
Terminal name:       RTA1:
Image name:          BISCO$DUA0:[SYS0.SYSCOMMON.][SYSEXE]AUTHORIZE.EXE
Object class name:   FILE
Object name:         SYS$SYSTEM:SYSUAF.DAT;4
User record:         NEWPORT
JTQUOTA:             New:        2048
                     Original:   1024
```

6.7.2.8. Contents of an Operator Log File

Example 6.7, "Sample Operator Log File (SYS$MANAGER:OPERATOR.LOG)" illustrates some typical messages found in an operator log file.

Example 6.7. Sample Operator Log File (SYS$MANAGER:OPERATOR.LOG)

```
%%%%%%%%%%  OPCOM, 19-APR-2016 22:26:07.90  %%%%%%%%%%
Device DMA0: is offline. ❶
Mount verification in progress.
%%%%%%%%%%  OPCOM, 19-APR-2016 22:26:20.22  %%%%%%%%%%
Mount verification completed for device DMA0:
%%%%%%%%%%  OPCOM, 19-APR-2016 22:33:54.07  %%%%%%%%%%
Operator '_ZEUS$VT333:' has been disabled, user JONES ❷
%%%%%%%%%%  OPCOM, 19-APR-2016 22:34:15.47  %%%%%%%%%%
Operator '_ZEUS$VT333:' has been enabled, user SMITH
%%%%%%%%%%  OPCOM, 19-APR-2016 22:34:15.57  %%%%%%%%%%
operator status for '_ZEUS$VT333:'
PRINTER, TAPES, DISKS, DEVICES
%%%%%%%%%%  OPCOM, 19-APR-2016 22:38:53.21  %%%%%%%%%%
request 1, from user PUBLIC ❸
Please mount volume KLATU in device MTA0:
The tape is in cabinet A
%%%%%%%%%%  OPCOM, 19-APR-2016 22:39:54.37  %%%%%%%%%%
request 1 was satisfied.
%%%%%%%%%%  OPCOM, 19-APR-2016 22:40:23.54  %%%%%%%%%%
message from user SYSTEM ❹
Volume "KLATU       " mounted, on physical device MTA0:
%%%%%%%%%%  OPCOM, 19-APR-2016 22:40:38.02  %%%%%%%%%%
request 2, from user PUBLIC
MOUNT new relative volume 2 () on MTA0:
%%%%%%%%%%  OPCOM, 19-APR-2016 22:41:07.54  %%%%%%%%%%
message from user SYSTEM Volume "KLATU       " dismounted, on physical
device MTA0:
15-APR-2016 22:42:14.81, request 2 completed by operator OPA0
```

```
 %%%%%%%%%%  OPCOM, 19-APR-2016 22:46:47.96  %%%%%%%%%%
request 4, from user PUBLIC
 _TTB5:, This is a sample user request with reply expected.
 %%%%%%%%%%  OPCOM, 19-APR-2016 22:47:38.50  %%%%%%%%%%
 request 4 was canceled
 %%%%%%%%%%  OPCOM, 19-APR-2016 22:48:21.15  %%%%%%%%%%
 message from user PUBLIC
 _TTB5:, This is a sample user request without a reply expected.
 %%%%%%%%%%  OPCOM, 19-APR-2016 22:49:37.64  %%%%%%%%%%
 Device DMA0: has been write locked.
 Mount verification in progress.
 %%%%%%%%%%  OPCOM, 19-APR-2016 23:33:54.07  %%%%%%%%%%
 message from user NETACP
 DECnet shutting down
```

The following messages appear in the example:

❶ Device status message

❷ Terminal enable and disable message

❸ User request and operator reply messages

❹ Volume mount and dismount messages

6.7.3. Setting Up the Operator Log File

The operator log file normally resides on the system disk in the [SYSMGR] directory. You can, however, maintain the log file in a different location by defining the logical name OPC$LOGFILE_NAME.

The size of and access to the OPERATOR.LOG file (or to the file pointed to by the logical OPC $LOGFILE_NAME) is limited by the size and access of the disk device on which it resides. If the disk device does not have enough room to write to the log file or if access to the device is restricted in any other way, records might be missing from the log file.

By default, the log file is created on all systems except for workstations in an OpenVMS Cluster environment (unless the workstation is the first system into the cluster). OPCOM determines whether a system is a workstation by testing the system for a graphics device. Specifically, this test is:

```
F$DEVICE (“*”, “WORKSTATION”, “DECW_OUTPUT”)
```

You can ensure that a log file will be created by defining the logical name OPC$LOGFILE_ENABLE to be true.

The system creates a new version of OPERATOR.LOG each time the system is rebooted. Note that one operator log file exists for each node; it is not a shared file.

Because this file is in ASCII format, you can print it. Print copies regularly and retain these copies for reference. *Section 6.7.5, "Printing the Operator Log File"* describes how to print copies of the operator log file.

6.7.3.1. Creating a New Version of the Operator Log File

You can use the DCL command REPLY/LOG to create a new version of the file at any time. The highest version is always the one in use and is inaccessible to other users. By default, messages of all operator classes are in the log file.

The following list contains guidelines for using the REPLY/LOG command:

- You can use the REPLY/LOG/ENABLE= *(keyword)* and REPLY/LOG/DISABLE= *(keyword)* commands to specify which operator classes to include in the log file.
- When you use the /LOG qualifier with the REPLY/ENABLE and REPLY/DISABLE commands, the classes you select in `keyword` are enabled or disabled for the log file rather than for the terminal.

If a log file is already open, the list of classes is preserved and enabled on the newly created log file. If a log file is not open, the value of the logical OPC$ENABLE_LOGFILE_CLASSES is used. If that logical does not exist, all classes are enabled on the new log file.

Note that if OPC$LOGFILE_CLASSES includes an invalid class then all classes are enabled, and a message similar to the following is displayed on the console at system startup and logged to the OPERATOR.LOG file:

```
%%%%%%%%%%% OPCOM 18-MAY-2002 13:28:33.12 %%%%%%%%%%%
"BADCLASS" is not a valid class name in OPC$LOGFILE_CLASSES
```

See *VSI OpenVMS System Manager's Manual, Volume 1: Essentials* for a description of the valid operator classes.

For more information, refer to the REPLY/LOG, REPLY/ENABLE, and REPLY/DISABLE commands in the *VSI OpenVMS DCL Dictionary*.

Example

The following command opens a log file to include messages about mounting and dismounting disks and tapes:

```
$ REPLY/LOG/ENABLE=(DISKS,TAPES)
```

6.7.3.2. Specifying Logical Names

You can specify the default state of the operator log files by defining logical names in the command procedure SYS$MANAGER:SYLOGICALS.COM. The following table lists these logical names and their functions. For more information about SYLOGICALS.COM, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

Caution

Setting the OPC$ALLOW_INBOUND and OPC$ALLOW_OUTBOUND logical names to FALSE severs all OPCOM traffic in the specified direction. All OPCOM messages, as well as any returned status messages that might be expected, will not be delivered.

Logical Name	Function
OPC $ALLOW_INBOUND	Allows OPCOM traffic that is inbound to the node to be turned on or off. By default, this logical name is set to TRUE. If this logical name is set to FALSE, the node will not receive any OPCOM messages from other nodes in the cluster.
OPC $ALLOW_OUTBOUND	Allows OPCOM traffic that is outbound from the node to be turned on or off. By default, this logical name is set to TRUE. If this logical name is set to FALSE, the node will not send any OPCOM messages to other nodes in the cluster.

Logical Name	Function
OPC $LOGFILE_ENABLE	Specifies whether an operator log file is opened. If defined to be true, an operator log file is opened. If defined to be false, no operator log file is opened. By default, a log file is opened on all systems except workstations in an OpenVMS Cluster environment.
OPC $LOGFILE_CLASSES	Specifies the operator classes that are enabled for the log file. By default, a log file is opened for all classes. The logical name can be a search list of the allowed classes, a comma-separated list, or a combination of the two. Note that you can define OPC$LOGFILE_CLASSES even if you do not define OPC$LOGFILE_ENABLE. In that case, the classes are used for any log files that are opened, but the default is used to determine whether to open the log file.
OPC$LOGFILE_NAME	Specifies the name of the log file. By default, the log file is named SYS $MANAGER:OPERATOR.LOG. If you specify a disk other than the system disk, include commands to mount that disk in the command procedure SYLOGICALS.COM.
OPC$OPA0_ENABLE	Overrides values of symbols for workstations in a cluster. If you define the logical as TRUE, it sets the OPA0 device to BROADCAST (overrides the NOBROADCAST default setting). For systems that are not workstations in a cluster, if you define the logical as FALSE, it sets the OPA0 device to NOBROADCAST.

Note

The only logical that is used for more than the initial startup of OPCOM is OPC$LOGFILE_NAME. All other OPCOM logicals are ignored. For example, a REPLY/LOG command opens a new operator log file even if the logical OPC$LOGFILE_ENABLE is defined to be false. To reset OPCOM states and classes after startup, use REPLY/ENABLE or REPLY/DISABLE commands.

6.7.4. Maintaining the Operator Log File

Devise a plan for regular maintenance of operator log files. One way is to start a new log file and rename the second-highest version daily. (See the example in the next section.) You might want to purge outdated versions of the operator log file on a regular basis. However, do not delete versions that you have not backed up. For more information, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

If OPCOM is inadvertently deleted, follow these steps to start it manually:

1. Log in to the SYSTEM account so that you have the required privileges to perform the operation.
2. Enter the following command to execute the startup command procedure (STARTUP.COM), specifying OPCOM as the command parameter:

```
$ @SYS$SYSTEM:STARTUP OPCOM
```

6.7.5. Printing the Operator Log File

Perform the following operation to produce a printed copy of the most recent version of the operator log file. (You must have OPER privilege.)

1. Use the following command to enable the terminal as an operator terminal:

   ```
   $ REPLY/ENABLE
   ```

2. Close the current log file and open a new one by entering the following command:

   ```
   $ REPLY/LOG
   ```

3. Set the default to SYS$MANAGER and enter the following command to list all versions of the file:

   ```
   $ SET DEFAULT SYS$MANAGER
   $ DIRECTORY OPERATOR.LOG
   ```

4. Rename the second-highest version to OPERATOR.OLD:

   ```
   $ RENAME OPERATOR.LOG;-1 OPERATOR.OLD
   ```

   The version number, –1, specifies that you want to rename the second-highest version of this file. (The highest version number is the current operator log file.)

5. Print the operator log file by entering the following command:

   ```
   $ PRINT OPERATOR.OLD
   ```

Example

```
$ REPLY/ENABLE ❶
$ REPLY/LOG ❷
%%%%%%%%%%  OPCOM, 19-APR-2016 12:28:20.11  %%%%%%%%%%
Logfile was closed by operator _MARS$VTA2: ❸
Logfile was HOMER::SYS$MANAGER:[SYSMGT]OPERATOR.LOG;27
%%%%%%%%%%  OPCOM, 19-APR-2016 12:29:24.52  %%%%%%%%%%
Logfile has been initialized by operator _MARS$VTA2:
Logfile is HOMER::SYS$MANAGER:[SYSMGT]OPERATOR.LOG;28
$ SET DEFAULT SYS$MANAGER ❹
$ DIRECTORY OPERATOR.LOG ❺
Directory SYS$MANAGER:[SYSMGT]
OPERATOR.LOG;28           OPERATOR.LOG;27
Total of 2 files.
$ RENAME OPERATOR.LOG;-1 OPERATOR.OLD ❻
$ PRINT OPERATOR.OLD ❼
```

The following list provides explanations of the numbered commands and responses in the example:

❶ The REPLY/ENABLE command enables the terminal as an operator terminal.

❷ The REPLY/LOG command closes the current log file and opens a new one.

❸ The response from OPCOM verifies that it has opened a new log file.

❹ The SET DEFAULT command sets the operator default disk to the system disk.

❺ The DIRECTORY command displays the files in the directory [SYSMGT] on the system disk.

❻ The RENAME command renames the second-highest version of the operator log file to OPERATOR.OLD.

❼ The PRINT command prints the old operator log file, OPERATOR.OLD.

6.8. Using Security Auditing

This section discusses how security auditing works; it also explains how to enable security auditing and how to create a new version of the security audit log file. For more information about the security audit log file, refer to the *VSI OpenVMS Guide to System Security*.

6.8.1. Understanding Security Auditing

Security auditing is the act of recording security-relevant events as they occur on the system. Security-relevant events are divided into a number of categories called **event classes**.

By default, the system enables security auditing when you install or upgrade your system for the events shown in *Table 6.11, "Event Classes Audited by Default"*.

Table 6.11. Event Classes Audited by Default

Class	Description
ACL	Access to any object holding a security Auditing ACE.
AUDIT	All uses of the SET AUDIT command. You cannot disable this category.
AUTHORIZATION	All changes to the authorization database: • System user authorization file (SYSUAF) • Network proxy authorization files: NETPROXY and NET$PROXY • Rights database (RIGHTSLIST)
BREAKIN	All break-in attempts: batch, detached, dialup, local, network, remote.
LOGFAILURE	All login failures: batch, dialup, local, remote, network, subprocess, detached.

If the security requirements at your site justify additional auditing, you can enable security auditing for other event classes by using the DCL command SET AUDIT, as explained in *Section 6.8.4, "Enabling Security Auditing for Additional Classes"*.

6.8.1.1. Security Audit Log File

The audit server process, which is created at system startup, records the events that are shown in *Table 6.11, "Event Classes Audited by Default"* in the security audit log file, SYS $MANAGER:SECURITY.AUDIT$JOURNAL.

The usefulness of the security audit log file depends upon the procedures you adopt to review the file on a regular basis. For example, you might implement the following procedure as part of your site audit review policy:

1. Create a new version of the security audit log file each morning.
2. Review the previous version of the log file for suspicious system activity. Depending on the number of security events you are auditing on your system, it might be impractical to review every audit record written to the audit log file. In that case, you might want to select a specific set of records from the log file (for example, all Authorization and Break in records, or all events created outside normal business hours).
3. If, during your review, you find any security events that appear suspicious, perform a more detailed inspection of the security audit log file, as described in the *VSI OpenVMS Guide to System Security*.

6.8.1.2. Audit Log Files in Mixed-Version Clusters

The Audit Analysis utility (ANALYZE/AUDIT) running on earlier-version systems is unable to process the current version of audit log files. You must use the current version of ANALYZE/AUDIT to process the current version of the audit log files. The recommended procedure is to maintain separate audit log files on mixed-version clusters.

If redirecting the audit log files, issue the following command on both the earlier-version node and on the node running the current version:

```
AUDIT/JOURNAL/DESTINATION=filespec
```

The destination file spec is stored in the audit server database file. By default, the files are stored in SYS$COMMON:[SYSMGR] and are called SECURITY_AUDIT.AUDIT$JOURNAL and SECURITY.AUDIT$JOURNAL, respectively.

The operating system allows workstations and other users with limited management resources to duplicate their audit log files on another node. The secondary log, a security archive file, is then available to a security administrator on a remote node who has the skills to analyze the file.

Each node in a cluster must have its own archive file. An archive file cannot be shared by multiple nodes in a cluster.

Refer to the *VSI OpenVMS Guide to System Security* for more information.

6.8.2. Displaying Security Auditing Information

To see which event classes your site currently audits, you can enter the DCL command SHOW AUDIT.

The follow example contains security information:

```
$ SHOW AUDIT

System security alarms currently enabled for:
  ACL
  Breakin:       dialup,local,remote,network,detached
  Privilege use:
    SECURITY
  Privilege failure:
    SECURITY

System security audits currently enabled for:
  ACL
  Authorization
  Breakin:       dialup,local,remote,network,detached
  Login:         dialup,local,remote,network,detached
  Logfailure:    batch,dialup,local,remote,network,subprocess,detached
  Logout:        dialup,local,remote,network,detached
  Privilege use:
  SECURITY
 Privilege failure:
  ACNT    ALLSPOOL  ALTPRI   AUDIT   BUGCHK  BYPASS  CMEXEC  CMKRNL
  DETACH  DIAGNOSE  EXQUOTA  GROUP   GRPNAM  GRPPRV  LOG_IO  MOUNT
  NETMBX  OPER      PFNMAP   PHY_IO  PRMCEB  PRMGBL  PRMMBX  PSWAPM
  READALL SECURITY  SETPRV   SHARE   SHMEM   SYSGBL  SYSLCK  SYSNAM
  SYSPRV  TMPMBX    VOLPRO   WORLD
 DEVICE access:
```

```
    Failure:     read,write,physical,logical,control
  FILE access:
    Failure:     read,write,execute,delete,control
  VOLUME access:
    Failure:     read,write,create,delete,control
```

6.8.3. Delaying Startup of Auditing

Ordinarily, the system turns on auditing in VMS$LPBEGIN just before SYSTARTUP_VMS.COM executes. You can change this behavior, however, by redefining the logical name SYS $AUDIT_SERVER_INHIBIT.

To change the point at which the operating system begins to deliver security-event messages, add the following line to the SYS$MANAGER:SYLOGICALS.COM command procedure:

```
$ DEFINE/SYSTEM/EXECUTIVE SYS$AUDIT_SERVER_INHIBIT YES
```

You can initiate auditing during another phase of system startup, perhaps at the end of SYSTARTUP_VMS.COM, by editing the command file to add the following line:

```
$ SET AUDIT/SERVER=INITIATE
```

For information about editing SYSTARTUP_VMS.COM, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

6.8.4. Enabling Security Auditing for Additional Classes

To enable security auditing for classes in addition to those shown in *Table 6.11, "Event Classes Audited by Default"*, use the following format:

```
SET AUDIT/ENABLE=keyword[,...] {/ALARM | /AUDIT}
```

The *VSI OpenVMS Guide to System Security* contains descriptions of event classes that you can enable.

When you enable auditing for additional event classes, you must specify two qualifiers:

1. /ENABLE
2. Either /ALARM or /AUDIT (Although you *must* specify one qualifier, you *can* specify both.)

The following table contains explanations of the /ENABLE, /ALARM, and /AUDIT qualifiers.

Qualifier	Explanation
/ENABLE	Defines which event classes you want audited. See *Chapter 7, "Tracking Resource Use"* for more information.
/ALARM /AUDIT	Defines the destination of the event message. • /ALARM directs the message to all enabled security operator terminals. • /AUDIT directs the message to the security audit log file. Use the /ALARM and /AUDIT qualifiers to report critical events. Less critical events can be written only to the security audit log file for later examination. The default event classes listed in *Table 6.11, "Event Classes Audited by Default"* are sent as both alarms and audits.

The system begins auditing new events on all nodes as soon as you enable them.

Examples

1. The command in the following example enables auditing for volume mounts and dismounts and sends messages to the security audit log file.

   ```
   $ SET AUDIT/ENABLE=MOUNT/AUDIT
   ```

2. The command in the following example enables auditing of unsuccessful file accesses and sends messages to all enabled security operator terminals as well as to the security audit log file.

   ```
   $ SET AUDIT/ALARM/AUDIT/ENABLE=ACCESS=FAILURE/CLASS=FILE
   ```

6.8.5. Disabling Security Auditing

The system continues auditing until you explicitly disable the classes with the /DISABLE qualifier using the following syntax:

```
SET AUDIT/DISABLE=keyword[,...] {/ALARM | /AUDIT}
```

6.8.6. Enabling a Terminal to Receive Alarm Messages

The system sends alarm messages to terminals enabled for security class messages. Security alarm messages are not written to the operator log file. They appear only on terminals enabled for security class messages.

In most cases, security alarm messages appear on the system console by default. Since messages scroll quickly off the screen, it is good practice to enable a separate terminal for security class messages and disable message delivery to the system console.

Either choose a terminal in a secure location that provides hardcopy output, or have dedicated staff to monitor the security operator terminal. You can enable any number of terminals as security operators.

To set up a terminal to receive security class alarms, enter the following DCL command from the designated terminal:

```
$ REPLY/ENABLE=SECURITY
```

Example

The following example shows a security alarm message:

```
%%%%%%%%%%  OPCOM  25-MAY-2016 16:07:09.20  %%%%%%%%%%
Message from user AUDIT$SERVER on GILMORE
Security alarm (SECURITY) on GILMORE, system id: 20300
Auditable event:          Process suspended ($SUSPND)
Event time:               25-MAY-2016 16:07:08.77
PID:                      30C00119
Process name:             Hobbit
Username:                 HUBERT
Process owner:            [LEGAL,HUBERT]
Terminal name:            RTA1:
Image name:               $99$DUA0:[SYS0.SYSCOMMON.][SYSEXE]SET.EXE
Status:                   %SYSTEM-S-NORMAL, normal successful completion
Target PID:               30C00126
Target process name:      SMISERVER
```

```
Target username:          SYSTEM
Target process owner:     [SYSTEM]
```

6.8.7. Generating Security Reports

The most common type of report to generate is a brief, daily listing of events. You can create a command procedure that runs in a batch job every evening before midnight to generate a report of the day's security event messages and send it to the system manager via Mail.

Note

Because the MOUNT command translates /NOLABEL to /FOREIGN in the audit record, use ANALYZE/AUDIT/SELECT=MOUNT_FLAGS=FOREIGN instead of ANALYZE/AUDIT/SELECT=MOUNT_FLAGS=NOLABEL.

The following example shows the ANALYZE/AUDIT command line you would use to generate this type of report:

```
$ ANALYZE/AUDIT/SINCE=TODAY/OUTPUT=31JAN2000.AUDIT -
_$ SYS$MANAGER:SECURITY.AUDIT$JOURNAL
$ MAIL/SUBJECT="Security Events" 31JAN2000.AUDIT SYSTEM
```

6.8.8. Creating a New Version of the Security Audit Log File

Because the security audit log file continues to grow until you take action, you must devise a plan for maintaining it.

Use the SET AUDIT command to create a new version of the clusterwide security audit log file. To prevent the loss of audit messages, the previous version of the audit log file is not closed until all audit messages stored in memory are written to the file.

6.8.8.1. Creating a New Clusterwide Version of the Log File

To open a new, clusterwide version of the security audit log file, use the following command:

```
$ SET AUDIT/SERVER=NEW_LOG
```

The audit server process opens a new version of the audit log file on each cluster node.

After you open the new log, rename the old version, using a naming convention for your files that incorporates in the file name a beginning or ending date for the data. Then copy the file to another disk, delete the log from the system disk to save space, and run the Audit Analysis utility on the old log.

By archiving this file, you maintain a clusterwide history of auditing messages. If you ever discover a security threat on the system, you can analyze the archived log files for a trail of suspicious user activity during a specified period of time.

6.8.8.2. Creating a New Node-Specific Version of the Log File

In some cases, OpenVMS Cluster nodes might not share the same system security audit log file. To create a new, node-specific version of the security audit log file, use the following commands:

```
$ SET AUDIT/DESTINATION=filespec
```

```
$ SET AUDIT/SERVER=NEW_LOG
```

where `filespec`is a logical name that points to a node-specific file; for example, SYS$SPECIFIC:[SYSMGR]SECURITY. System security audit log files on other nodes are unaffected.

6.9. Monitoring Operating System Performance

The Monitor utility (MONITOR) is a system management tool that you can use to obtain information about operating system performance. Various MONITOR qualifiers collect system performance data from the running system or play back data recorded previously in a recording file. When you play back data, you can display it, summarize it, and even rerecord it to reduce the amount of data in the recording file.

Following an explanation of the Monitor utility are sections that tell how to perform these tasks:

Task	**Section**
Invoking the Monitor utility	*Section 6.9.2, "Invoking MONITOR"*
Using live display monitoring	*Section 6.9.3, "Using Live Display Monitoring"*
Using live recording monitoring	*Section 6.9.4, "Using Live Recording Monitoring"*
Using concurrent display and recording monitoring	*Section 6.9.5, "Using Concurrent Display and Recording Monitoring"*
Using playback monitoring	*Section 6.9.6, "Using Playback Monitoring"*
Using remote playback monitoring	*Section 6.9.7, "Using Remote Playback Monitoring"*
Rerecording monitoring	*Section 6.9.8, "Rerecording Monitoring"*
Running MONITOR continuously	*Section 6.9.9, "Running MONITOR Continuously"*
Using remote monitoring	*Section 6.9.10, "Using Remote Monitoring"*

For additional information about interpreting the information the Monitor utility provides, refer to the *OpenVMS Performance Management Manual*. For additional information about using the Monitor utility, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

6.9.1. Understanding MONITOR

Using MONITOR, you can monitor classes of systemwide performance data (such as system I/O statistics, page management statistics, and time spent in each of the processor modes) at specifiable intervals, and produce several types of output. You can also develop a database of performance information for your system by running MONITOR continuously as a background process, as explained in *Section 6.9.9, "Running MONITOR Continuously"*.

6.9.1.1. MONITOR Classes

Each MONITOR class consists of data items that, taken together, provide a statistical measure of a particular system performance category. The data items defined for individual classes are listed in the description of the MONITOR command in the *VSI OpenVMS System Management Utilities Reference Manual*.

To monitor a particular class of information, specify a class name on the MONITOR command line. The information MONITOR displays depends on the type of class you select. *Table 6.12, "Types of MONITOR Classes"* compares the two MONITOR class types.

Table 6.12. Types of MONITOR Classes

Type of class	Description
System	Provides statistics on resource use for the entire system
Component	Provides statistics on the contribution of individual components to the overall system or cluster

As an example of the distinction between types of MONITOR classes, the IO class includes a data item to measure all direct I/O operations for the entire system, and is therefore a system class. The DISK class measures direct I/O operations for individual disks, and is therefore a component class.

Table 6.13, "MONITOR Classes" describes each MONITOR class and indicates whether it is a system or component class.

Table 6.13. MONITOR Classes

Class	Type	Description
ALL_CLASSES	System or Component	Statistics for all classes
CLUSTER	System	Clusterwide performance statistics
DECNET	System	DECnet for OpenVMS statistics
DISK	Component	Disk I/O statistics
DLOCK	System	Distributed lock management statistics
FCP	System	File control primitive statistics
FILE_SYSTEM_CACHE	System	File system cache statistics
IO	System	System I/O statistics

Class	Type	Description
LOCK	System	Lock management statistics
MODES	Component	Time spent in each of the processor modes
MSCP_SERVER	System	MSCP server statistics
PAGE	System	Page management statistics
PROCESSES	Component	Statistics on all processes
RMS	Component	Record Management Services statistics
SCS	Component	System Communications Services statistics
STATES	System	Number of processes in each of the scheduler states
SYSTEM	System	Summary of statistics from other classes
TRANSACTION	System	DECdtm services statistics
VBS (VAX specific)	System	Virtual balance slot statistics
VECTOR	System	Vector processor scheduled usage

6.9.1.2. Display Data

Except in the PROCESSES class, all data item statistics are displayed as rates or levels:

- Rates are shown in number of occurrences per second.
- Levels are values that indicate the size of the monitored data item.

You can request any or all of four different statistics for each data item:

Statistic	Description
Current rate or level	Most recently collected value for the rate or level
Average rate or level	Measured from the beginning of the MONITOR request
Minimum rate or level	Measured from the beginning of the MONITOR request
Maximum rate or level	Measured from the beginning of the MONITOR request

For the DISK, MODES, SCS, and STATES classes, you can optionally express all statistics as percentages.

In the PROCESSES class, MONITOR displays descriptive information, level information, and counters that increase over time.

6.9.1.3. Output Types

MONITOR collects system performance data by class and produces three forms of optional output, depending on the qualifier you specify:

Qualifier	Description
/DISPLAY	Produces output in the form of ASCII screen images, which are written at a frequency governed by the /VIEWING_TIME qualifier.
/RECORD	Produces a binary recording file containing data collected for requested classes; one record for each class is written per interval.

Qualifier	Description
/SUMMARY	Produces an ASCII file containing summary statistics for all requested classes over the duration of the MONITOR request.

If you specify /INPUT with any of these qualifiers, MONITOR collects performance data from one or more previously created recording files; otherwise, data is collected from counters and data structures on the running system.

You use the /BEGINNING and /ENDING qualifiers to specify, respectively, when you want a MONITOR request to begin and end.

Using the /DISPLAY Qualifier

Information collected by MONITOR is normally displayed as ASCII screen images. You can use the optional /DISPLAY qualifier to specify a disk file to contain the information. If you omit the file specification, output is directed to SYS$OUTPUT.

Note

Be careful when you use the /DISPLAY qualifier. Because MONITOR enters display information into the file continuously, its size can grow very quickly.

Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for a discussion of the /DISPLAY qualifier.

Using the /RECORD Qualifier

When you use the /RECORD qualifier, all data pertaining to the class is recorded, even if you are concurrently displaying only a single statistic or a single item of a component statistics class. The file is created when a MONITOR request is initiated and closed when a request terminates. You can use the resulting file as a source file for later requests to format and display the data on a terminal, to create a summary file, or to create a new recording file with different characteristics.

6.9.2. Invoking MONITOR

To invoke the Monitor utility, enter the following DCL command:

```
$ MONITOR
```

MONITOR then displays the following prompt:

```
MONITOR>
```

In response to the prompt, you can enter any of the MONITOR commands, which are described in *VSI OpenVMS System Management Utilities Reference Manual*. The most frequently used MONITOR command, however, specifies a class name.

Example

```
MONITOR>MONITOR PAGE
```

In this example, you specify the PAGE class name in the MONITOR command to monitor page management statistics.

You can also use the MONITOR command from DCL command level.

How to Override or Terminate a MONITOR Request

Generally, each MONITOR request runs until the time specified or implied by the /ENDING qualifier. However, to override or terminate a MONITOR request, you can press one of the following combinations of keys:

Keys	Description
Ctrl/W	Temporarily overrides a /VIEWING_TIME value and generates a new display immediately following the current one. This feature is useful when a broadcast message overwrites the MONITOR display area. You can also use Ctrl/W in conjunction with a large /VIEWING_TIME value to generate display events on demand.
Ctrl/C	Terminates the current request without exiting from the utility. You can then initiate a new request or enter any MONITOR command at the MONITOR> prompt.
Ctrl/Z	Terminates the current request and exits from MONITOR.

6.9.3. Using Live Display Monitoring

Use the live display monitoring mode of operation when you want to examine the activity of a running system, either on a routine basis or as part of an installation checkout, tuning, or troubleshooting exercise. The system does not keep a historical record of output. The following examples show how to use the live display monitoring mode.

Examples

1. `$ MONITOR PROCESSES/TOPCPU`

   The command displays a bar graph showing the eight processes that were the top consumers of CPU time during the period between displays. It also displays the amount of CPU time each process used.

   The command might produce a display similar to the following one:

```
                      OpenVMS Monitor Utility
                       TOP CPU TIME PROCESSES
                            on node BOMBAY
                         20-JAN-2016 10:06:49

                                  0         25        50        75
 100
                                  + - - - - + - - - - - + - - - - + - - - - - -
+
07E00181  CAFARET          100  ****************************************
                                  |         |         |         |
 |
                                  |         |         |         |
 |
                                  |         |         |         |
 |
                                  |         |         |         |
 |
                                  |         |         |         |
 |
```

```
                                    |          |          |          |
  |

                                   + - - - - + - - - - + - - - - + - - - - - -
+
```

This example shows that user CAFARET is using 100 percent of the CPU time available. To display more information about the computer resources a user is using, use a command similar to the following one:

```
$ SHOW PROCESS/CONTINUOUS/ID=07E00181
```

For this example, the most useful information in the resulting display is the name of the image at the end of the display; for example:

```
.
.
.
$1$DUA1:[SYS1D.SYSCOMMON.][SYSEXE]RODAN.EXE
```

This example indicates that CAFARET is running RODAN.EXE, which might be new software that is unintentionally running in a loop. This situation would occur if CAFARET were a privileged user running a process at a higher priority than other users.

2. `$ MONITOR/DISPLAY=PROCESSES.LOG PROCESSES`

   You can route MONITOR display output to any supported terminal device or to a disk file. This command writes MONITOR's display process statistics to the file PROCESSES.LOG. You can then print this file on a hardcopy device.

   Caution

   Because data is continuously added to the display file, be careful that the file does not grow too large.

3. ```
   $ MY_CLASSES :== -
   _$ "DECNET+FCP+IO+LOCK+MODES+PAGE+PROCESSES+STATES"
   $ MONITOR/NODE=(CURLEY,LARRY)/INTERVAL=20/VIEWING_TIME=8 'MY_CLASSES'
   ```

   You might find it convenient to establish DCL symbols for frequently used combinations of class names, as in this example. The MONITOR command collects selected classes of data for OpenVMS Cluster nodes CURLEY and LARRY every 20 seconds. Every 8 seconds, the system displays the most recently collected data for one of the classes. MONITOR predetermines the ordering of the classes for display.

6.9.4. Using Live Recording Monitoring

Use live recording to capture MONITOR data for future use. Possible uses include the following ones:

- Installation checkout, tuning, troubleshooting; that is, all the uses that are listed for live display monitoring.

  Choose recording over display when you want to capture more classes than you can reasonably watch at a terminal, when a terminal is not available, or when you want to gather data about the system but cannot spend time at the terminal until later.
   ```

- Routine performance data gathering for long-term analysis.

  You can record MONITOR data on a routine basis and summarize it to gather data about system resource use over long periods of time.

  Caution

  Because data is continuously added to the recording file, be careful that the file does not grow too large.

The following example shows how to use the live recording mode of operation.

Example

```
$ MONITOR/NODE=(LARRY,MOE)/NODISPLAY/RECORD MODES+STATES
```

The command in this example records data on the time spent in each processor mode and on the number of processes in each scheduler state for nodes LARRY and MOE. The command does not display this information.

6.9.5. Using Concurrent Display and Recording Monitoring

Use the concurrent display and recording mode of operation when you want to both retain performance data and watch as it is being collected. Because MONITOR allows shared read access to the recording file, a separate display process can play back the recording file as it is being written by another process.

The following examples show how to use the concurrent display and recording mode of operation. The first example both collects and records data in the same command. The second and third examples show how you can perform concurrent recording and display using two separate processes: the process in the second example performs recording; the process in the third example plays back the file to obtain a summary.

Example

1. ```
   $ MONITOR/RECORD FCP/AVERAGE,FILE_SYSTEM_CACHE/MINIMUM
   ```

   This command collects and records file system data and file system cache data every 3 seconds. It also displays, in bar graph form, average FCP statistics and minimum FILE_SYSTEM_CACHE statistics. The display alternates between the two graphs every 3 seconds. You can obtain current statistics in a subsequent playback request.

2. ```
   $ MONITOR/RECORD=SYS$MANAGER:ARCHIVE.DAT -
   _$ /INTERVAL=300/NODISPLAY ALL_CLASSES
   ```

   This command archives data for all classes once every 5 minutes. You might find it convenient to execute a similar command in a batch job, taking care to monitor disk space usage.

3. ```
   $ MONITOR/INPUT=SYS$MANAGER:ARCHIVE.DAT: -
   _$ /NODISPLAY/SUMMARY/BEGINNING="-1" PAGE,IO
   ```

   The command in this example produces a summary of page and I/O activity that occurred during the previous hour, perhaps as part of an investigation of a reported performance problem. Note that because the recording process executes an OpenVMS RMS flush operation every 5 minutes, up to 5
   ```

minutes of the most recently collected data is not available to the display process. You can specify the time between flush operations explicitly with the /FLUSH_INTERVAL qualifier. Note also that the display process must have read access to the recording file.

6.9.6. Using Playback Monitoring

Use playback of a recording file to obtain terminal output and summary reports of all collected data or a subset of it. You can make a subset of data according to class, node, or time segment. For example, if you collect several classes of data for an entire day, you can examine or summarize the data on one or more classes during any time period in that day.

You can also display or summarize data with a different interval than the one at which it was recorded. You control the actual amount of time between displays of screen images with the /VIEWING_TIME qualifier. The following examples show how to use the playback mode of operation.

Example

1.
```
$ MONITOR/RECORD/INTERVAL=5 IO
    .
    .
    .
$ MONITOR/INPUT IO
```

   The commands in this example produce system I/O statistics. The first command gathers and displays data every 5 seconds, beginning when you enter the command and ending when you press Ctrl/Z. In addition, the first command records binary data in the default output file MONITOR.DAT. The second command plays back the I/O statistics display, using the data in MONITOR.DAT for input. The default viewing time for the playback is 3 seconds, but each screen display represents 5 seconds of monitored I/O statistics.

2.
```
$ MONITOR/RECORD/NODISPLAY -
_$ /BEGINNING=08:00:00 -
_$ /ENDING=16:00:00 -
_$ /INTERVAL=120 DISK
    .
    .
    .
$ MONITOR/INPUT/DISPLAY=HOURLY.LOG/INTERVAL=3600 DISK
```

   The sequence of commands in this example illustrates data recording with a relatively small interval and data playback with a relatively large interval. This is useful for producing average, minimum, and maximum statistics that cover a wide range of time, but have greater precision than if they had been gathered using the larger interval. The first command records data on I/O operations for all disks on the system for the indicated 8-hour period, using an interval of 2 minutes. The second command plays the data back with an interval of 1 hour, storing display output in the file HOURLY.LOG. You can then type or print this file to show the cumulative average disk use at each hour throughout the 8-hour period.

   ### Note

   The current statistic in HOURLY.LOG shows the current data in terms of the original collection interval of 120 seconds, not the new collection interval of 3600 seconds.

3.
```
$ MONITOR/INPUT/NODISPLAY/SUMMARY=DAILY.LOG DISK
```

The command in this example uses the recording file created in the previous example to produce a one-page summary report file showing statistics for the indicated 8-hour period. The summary report has the same format as a screen display. For example:

```
                    OpenVMS Monitor Utility
                      DISK I/O STATISTICS
                        on node TLC     From: 25-JAN-2016 08:00:00
                           SUMMARY      To:   25-JAN-2016 16:00:00

I/O Operation Rate                CUR     AVE     MIN     MAX

DSA0:                 SYSTEM_0   0.53    1.50    0.40    3.88
DSA1:                 SYSTEM_1   0.00    0.39    0.00    8.38
DSA4:                 WORK_0     0.00    0.11    0.00    1.29
DSA5:                 WORK_1     0.03    0.87    0.00    5.95
DSA6:                 WORK_2     0.03    0.25    0.00    2.69
DSA7:                 WORK_3     0.04    0.97    0.00   20.33
DSA17:                TOM_DISK   0.00    0.04    0.00    0.80
DSA23:                MKC        0.00    0.00    0.00    0.13
$4$DUA0:    (RABBIT)  SYSTEM_0   0.20    0.65    0.17    1.97
$4$DUA2:    (RABBIT)  SYSTEM_0   0.20    0.65    0.17    1.97
$4$DUA3:    (RABBIT)  SYSTEM_1   0.00    0.14    0.00    2.49

 PLAYBACK                          SUMMARIZING
```

6.9.7. Using Remote Playback Monitoring

If suitably privileged, you can collect MONITOR data from any system to which your system has a DECnet connection. You can then display the data live on your local system. To do so, follow these steps:

1. In the default DECnet directory on each remote system, create a file named MONITOR.COM, similar to the following example:

   ```
   $ !
   $ !    * Enable MONITOR remote playback *
   $ !
   $ MONITOR /NODISPLAY/RECORD=SYS$NET ALL_CLASSES
   ```

2. On your local system, define a logical name for the remote system from which you want to collect data. Use the following syntax:

   ```
   DEFINE remotenodename_mon node::task=monitor
   ```

   You might want to define, in a login command procedure, a series of logical names for all the systems you want to access.

3. To display the remote MONITOR data as it is being collected, enter a command using the following syntax:

   ```
   MONITOR/INPUT=remotenodename_mon classnames
   ```

You can also place MONITOR.COM files in directories other than the default DECnet directory and use access control strings or proxy accounts to invoke these command files remotely.

When you invoke MONITOR on your local system, a process is created on the remote system that executes the MONITOR.COM command file. The remote system therefore experiences some associated

CPU and DECnet overhead. You can regulate the overhead in the MONITOR.COM file by using the /INTERVAL qualifier and the list of class names.

Section 6.9.10, "Using Remote Monitoring" describes remote monitoring in a mixed-version cluster system.

6.9.8. Rerecording Monitoring

Rerecording is a combination of playback and recording. You can use it for data reduction of recording files. When you play back an existing recording file, all MONITOR options are available to you; thus, you can choose to record a subset of the recorded classes and a subset of the recorded time segment and a larger interval value.

All these techniques produce a new, smaller recording file at the expense of some of the recorded data. A larger interval value reduces the volume of the collected data, so displays and summary output produced from the newer recorded file will be less precise. Note that average rate values are not affected in this case, but average level values are less precise (since the sample size is reduced), as are maximum and minimum values. The following example shows how to use the rerecording mode of operation:

Example

```
$ SUBMIT MONREC.COM
```

MONREC.COM contains the following commands:

```
$ MONITOR/NODISPLAY/RECORD/INTERVAL=60 /BEGINNING=8:00/ENDING=16:00
 DECNET,LOCK
$ MONITOR/INPUT/NODISPLAY/RECORD DECNET
```

The first command runs in a batch job, recording DECnet and lock management data once every minute between the hours of 8 A.M. and 4 P.M.. The second command, which is issued after the first command completes, rerecords the data by creating a new version of the MONITOR.DAT file, containing only the DECnet data.

6.9.9. Running MONITOR Continuously

You can develop a database of performance information for your system by running MONITOR continuously as a background process. This section contains examples of procedures that you, as cluster manager, might use to create multifile clusterwide summaries.

You can adapt the command procedures to suit conditions at your site. Note that you must define the logical names SYS$MONITOR and MON$ARCHIVE in SYSTARTUP.COM before executing any of the command files.

The directory with the logical name SYS$EXAMPLES includes three command procedures that you can use to establish the database. Instructions for installing and running the procedures are in the comments at the beginning of each procedure. *Table 6.14, "MONITOR Command Procedures"* contains a brief summary of these procedures.

Table 6.14. MONITOR Command Procedures

Procedure	Description
MONITOR.COM	Creates a summary file from the recording file of the previous boot, and then begins recording for this boot. The recording interval is 10 minutes.

Procedure	Description
MONSUM.COM	Generates two clusterwide multifile summary reports that are mailed to the system manager: one report is for the previous 24 hours, and the other is for the previous day's prime-time period (9 A.M. to 6 P.M.). The procedure resubmits itself to run each day at midnight.
SUBMON.COM	Starts MONITOR.COM as a detached process. Invoke SUBMON.COM from the site-specific startup command procedure.

While MONITOR records data continuously, a summary report can cover any finite time segment. The MONSUM.COM command procedure, which is executed every midnight, produces and mails the two multifile summary reports described in *Table 6.14, "MONITOR Command Procedures"*. Because these reports are not saved as files, to keep them, you must either extract them from your mail file or alter the MONSUM.COM command procedure to save them.

6.9.9.1. Using the MONITOR.COM Procedure

The procedure in *Example 6.8, "MONITOR.COM Procedure "* archives the recording file and summary file from the previous boot and initiates continuous recording for the current boot. (Note that this procedure does not purge recording files.)

Example 6.8. MONITOR.COM Procedure

```
$ SET VERIFY
$ !
$ !  MONITOR.COM
$ !
$ !  This command file is to be placed in a cluster-accessible directory
$ !  called SYS$MONITOR and submitted at system startup time as a detached
$ !  process via SUBMON.COM. For each node, MONITOR.COM creates, in
$ !  SYS$MONITOR, a MONITOR recording file that is updated throughout the
$ !  life of the boot. It also creates, in MON$ARCHIVE, a summary file
$ !  from the recording file of the previous boot, along with a copy of
 that
$ !  recording file. Include logical name definitions for both cluster-
$ !  accessible directories, SYS$MONITOR and MON$ARCHIVE, in SYSTARTUP.COM.
$ !
$ SET DEF SYS$MONITOR
$ SET NOON
$ PURGE MONITOR.LOG/KEEP:2
$ !
$ !  Compute executing node name and recording and summary file names
$ !  (incorporating node name and date).
$ !
$ NODE = F$GETSYI("NODENAME")
$ SEP = ""
$ IF NODE .NES. "" THEN SEP = "_"
$ DAY = F$EXTRACT (0,2,F$TIME())
$ IF F$EXTRACT(0,1,DAY) .EQS. " " THEN DAY = F$EXTRACT(1,1,DAY)
$ MONTH = F$EXTRACT(3,3,F$TIME())
$ ARCHFILNAM = "MON$ARCHIVE:"+NODE+SEP+"MON"+DAY+MONTH
$ RECFIL = NODE+SEP+"MON.DAT"
$ SUMFIL = ARCHFILNAM+".SUM"

$ !
$ !  Check for existence of recording file from previous boot and skip
$ !  summary if not present.
```

```
$ !
$ OPEN/READ/ERROR=NORECFIL RECORDING 'RECFIL'
$ CLOSE RECORDING
$ !
$ !
$ !  Generate summary file from previous boot.
$ !
$ MONITOR /INPUT='RECFIL' /NODISPLAY /SUMMARY='SUMFIL' -
$ ALL_CLASSES+MODE/ALL+STATES/ALL+SCS/ITEM=ALL+SYSTEM/ALL+DISK/ITEM=ALL
$ !
$ !
$ !  Compute subject string and mail summary file to cluster manager.
$ !
$ !
$ A="""
$ B=" MONITOR Summary "
$ SUB = A+NODE+B+F$TIME()+A
$ MAIL/SUBJECT='SUB' 'SUMFIL' CLUSTER_MANAGER
$ !
$ !
$ !  Archive recording file and delete it from SYS$MONITOR.
$ !
$ COPY 'RECFIL' 'ARCHFILNAM'.DAT
$ DELETE 'RECFIL';*
$ !
$ NORECFIL:
$ SET PROCESS/PRIORITY=15
$ !
$ !
$ !  Begin recording for this boot.  The specified /INTERVAL value is
$ !  adequate for long-term summaries; you might need a smaller value
$ !  to get reasonable "semi-live" playback summaries (at the expense
$ !  of more disk space for the recording file).
$ !
$ MONITOR /INTERVAL=300 /NODISPLAY /RECORD='RECFIL' ALL_CLASSES
$ !
$ !
$ !  End of MONITOR.COM
$ !
```

6.9.9.2. Using the SUBMON.COM Procedure

The procedure in *Example 6.9, "SUBMON.COM Procedure "* submits MONITOR.COM as a detached process from SYSTARTUP.COM to initiate continuous recording for the current boot.

Example 6.9. SUBMON.COM Procedure

```
$ SET VERIFY
$ !
$ !  SUBMON.COM
$ !
$ !  This command file is to be placed in a cluster-accessible directory
$ !  called SYS$MONITOR.  At system startup time, for each node, it is
$ !  executed by SYSTARTUP.COM, following logical name definitions for
$ !  the cluster-accessible directories SYS$MONITOR and MON$ARCHIVE.
$ !
$ !
$ !  Submit detached MONITOR process to do continuous recording.
```

```
$ !
$ !
$ RUN   SYS$SYSTEM:LOGINOUT.EXE -
        /UIC=[1,4]               -
        /INPUT=SYS$MONITOR:MONITOR.COM  -
        /OUTPUT=SYS$MONITOR:MONITOR.LOG -
        /ERROR=SYS$MONITOR:MONITOR.LOG  -
        /PROCESS_NAME="Monitor" -

        /WORKING_SET=512 -
        /MAXIMUM_WORKING_SET=512 -
        /EXTENT=512/NOSWAPPING
$ !
$ !
$ !  End of SUBMON.COM
$ !
```

6.9.9.3. Using the MONSUM.COM Procedure

The procedure in *Example 6.10, "MONSUM.COM Procedure "* produces daily and prime-time clusterwide summaries.

Example 6.10. MONSUM.COM Procedure

```
$ SET VERIFY
$ !
$ !  MONSUM.COM
$ !
$ !  This command file is to be placed in a cluster-accessible directory
$ !  called SYS$MONITOR and executed at the convenience of the cluster
$ !  manager.  The file generates both 24-hour and "prime time" cluster
$ !  summaries and resubmits itself to run each day at midnight.
$ !
$ SET DEF SYS$MONITOR
$ SET NOON
$ !
$ !  Compute file specification for MONSUM.COM and resubmit the file.
$ !
$ FILE = F$ENVIRONMENT("PROCEDURE")
$ FILE = F$PARSE(FILE,,,"DEVICE")+F$PARSE(FILE,,,"DIRECTORY")+F
$PARSE(FILE,,,"NAME")
$ SUBMIT 'FILE' /AFTER=TOMORROW /NOPRINT
$ !
$ !  Generate 24-hour cluster summary.
$ !
$ !
$ MONITOR/INPUT=(SYS$MONITOR:*MON*.DAT;*,MON$ARCHIVE:*MON*.DAT;*) -
  /NODISPLAY/SUMMARY=MONSUM.SUM -
  ALL_CLASSES+DISK/ITEM=ALL+SCS/ITEM=ALL-
  /BEGIN="YESTERDAY+0:0:0.00" /END="TODAY+0:0:0.00" /BY_NODE
$ !
$ !
$ !  Mail 24-hour summary file to cluster manager and delete the file from
$ !  SYS$MONITOR.
$ !
$ !
$ MAIL/SUBJECT="Daily Monitor Clusterwide Summary" MONSUM.SUM
 CLUSTER_MANAGER
```

```
$ DELETE MONSUM.SUM;*
$ !
$ !  Generate prime-time cluster summary.
$ !
$ !
$ MONITOR/INPUT=(SYS$MONITOR:*MON*.DAT;*,MON$ARCHIVE:*MON*.DAT;*) -
  /NODISPLAY/SUMMARY=MONSUM.SUM -
  ALL_CLASSES+DISK/ITEM=ALL+SCS/ITEM=ALL-
  /BEGIN="YESTERDAY+9:0:0.00" /END="YESTERDAY+18:0:0.00" /BY_NODE
$ !
$ !
$ !  Mail prime-time summary file to cluster manager and delete the file
$ !  from SYS$MONITOR.
$ !
$ !
$ MAIL/SUBJECT="Prime-Time Monitor Clusterwide Summary" MONSUM.SUM
 CLUSTER_MANAGER
$ DELETE MONSUM.SUM;*
$ !
$ !  End of MONSUM.COM
$ !
```

Note that Mail commands in this procedure send files to user CLUSTER_MANAGER. Replace CLUSTER_MANAGER with the appropriate user name or logical name for your site.

Because summary data might be extensive, VSI recommends that you print out summary files.

6.9.10. Using Remote Monitoring

MONITOR is capable of using both TCP/IP and DECnet as a transport mechanism. Beginning with OpenVMS Version 7.0, to use TCP/IP, you must start the TCP/IP server by issuing the following command inside SYS$STARTUP:SYSTARTUP_VMS.COM:

```
$ @SYS$STARTUP:VPM$STARTUP.COM
```

DECnet continues to work as in the past: a network object is created at the time of the request.

Remote Monitoring in a Mixed-Version OpenVMS Cluster System

You can monitor any node in an OpenVMS Cluster system either by issuing the MONITOR CLUSTER command or by adding the /NODE qualifier to any interactive MONITOR request.

Remote monitoring in an OpenVMS Cluster system might not be compatible, however, between nodes that are running different versions of OpenVMS. *Table 6.15, "Remote Monitoring Compatibility in an OpenVMS Cluster System"* shows the compatibility of versions for remote monitoring.

Table 6.15. Remote Monitoring Compatibility in an OpenVMS Cluster System

Versions	**OpenVMS Alpha and VAX Version 6.*n* or 7.*n***	**OpenVMS Alpha Version 1.5 and VAX Version 5.*n***
OpenVMS Alpha and VAX Version 6. *n* or 7. *n*	Yes	No

Versions	**OpenVMS Alpha and VAX Version 6.*n* or 7.*n***	**OpenVMS Alpha Version 1.5 and VAX Version 5.*n***
OpenVMS Alpha Version 1.5 and VAX Version 5. *n*	No	Yes

If you attempt to monitor a remote node that is incompatible, the system displays the following message:

```
%MONITOR-E-SRVMISMATCH, MONITOR server on remote node is an incompatible
 version
```

If this is the case, contact your VSI support representative for a remedial kit that corrects this problem. Before you install the remedial kit, you can still use MONITOR to obtain data about the remote node. To do this, record the data on the remote node and then run the MONITOR playback feature to examine the data on the local node.

Another difference exists when you monitor remote nodes in an OpenVMS Cluster system. Beginning with OpenVMS Version 6.2, the limit on the number of disks that can be monitored was raised from 799 to 909 for record output and from 799 to 1817 for display and summary outputs. If you monitor a remote node running OpenVMS Version 6.2 or later from a system running a version earlier than OpenVMS Version 6.2, the old limit of 799 applies.

For more information about MONITOR, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

Chapter 7. Tracking Resource Use

This chapter describes how to find out how your system resources have been used. You can use this information to:

- Charge users for the resources they have used. You can produce reports of the resources used by individual users.
- Plan your future equipment requirements. You can monitor changing patterns of resource use and predict future demands.
- Troubleshoot the system. You can check the final exit status of processes.
- Improve system performance. You can find out the load that individual images and processes place on your system.
- Detect security breaches. You can identify unusual patterns of resource use.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Determining which resources are being tracked	*Section 7.2, "Determining Which Resources Are Being Tracked "*
Controlling which resources are tracked	*Section 7.3, "Controlling Which Resources Are Tracked"*
Starting up a new accounting file	*Section 7.4, "Starting Up a New Accounting File"*
Moving the accounting file	*Section 7.5, "Moving the Accounting File"*
Producing reports of resource use	*Section 7.6, "Producing Reports of Resource Use"*
Setting up accounting groups	*Section 7.7, "Setting Up Accounting Groups "*
Monitoring disk space	*Section 7.8, "Monitoring Disk Space"*

This chapter explains the following concept:

Concept	Section
Accounting files	*Section 7.1, "Understanding Accounting Files"*

7.1. Understanding Accounting Files

The system gathers information about resource use. For example, the information can include the resources such as CPU time used by each print job. The system stores this information in **accounting files**.

The resources tracked by default depend on the model of computer you use. However, you can control which resources are tracked. If you do not want to track resource use, you can stop the accounting file tracking resource use altogether. (See *Section 7.3, "Controlling Which Resources Are Tracked"*.)

Each node in an OpenVMS Cluster has its own accounting file, known as its **current accounting file**. By default, this file is SYS$MANAGER:ACCOUNTNG.DAT, but you can control which file is used (see *Section 7.5, "Moving the Accounting File"*).

The information in the accounting files is in binary. You cannot display it with the TYPE command. To display the information, use the Accounting utility (ACCOUNTING). (See *Section 7.6, "Producing Reports of Resource Use"*.)

7.2. Determining Which Resources Are Being Tracked

To determine which resources are currently being tracked, use the SHOW ACCOUNTING command:

```
$ SHOW ACCOUNTING
```

This command produces a screen display (see the example) that contains keywords in the following two categories:

- Keywords that show which types of resource are being tracked:

Keyword	Type of Resource
IMAGE	Resources used by an image
LOGIN_FAILURE	Resources used by an unsuccessful attempt to log in
MESSAGE	Unformatted resource record written to the accounting file by a call to the $SNDJBC system service
PRINT	Resources used by a print job
PROCESS	Resources used by a process

- Keywords that show which types of process are being tracked. When the resources for processes or images are tracked, these keywords show the process type:

Keyword	Type of Process
BATCH	Batch process
DETACHED	Detached process

Keyword	Type of Process
INTERACTIVE	Interactive process
NETWORK	Network process
SUBPROCESS	Subprocess (the parent process can be a batch, detached, interactive, or network process)

Example

```
$ SHOW ACCOUNTING
Accounting is currently enabled to log the following activities:
      PROCESS        any process termination
      IMAGE          image execution
      INTERACTIVE    interactive job termination
      LOGIN_FAILURE  login failures
      NETWORK        network job termination
      PRINT          all print jobs
```

The keywords in this example show that the local node is tracking the resources used by each:

- Interactive and network process
- Image running in an interactive or network process
- Login failure
- Print job

7.3. Controlling Which Resources Are Tracked

You can control which resources the system tracks. To save disk space, you can stop the system tracking resources you are not interested in.

How to Perform This Task

1. Use the SET ACCOUNTING command with the /ENABLE and /DISABLE qualifiers in the following format to control which resources are tracked:

   ```
   SET ACCOUNTING/DISABLE[=(keyword[,...])]/ENABLE[=(keyword[,...])]
   ```

   The keywords are the same as those explained in *Section 7.2, "Determining Which Resources Are Being Tracked "*.

2. To make this change permanent, edit the SET ACCOUNTING command in the SYS $MANAGER:SYSTART_VMS.COM startup file.

Example

This example prevents the tracking of all resources except those used by interactive and batch processes:

```
$ SET ACCOUNTING/DISABLE/ENABLE=(PROCESS,INTERACTIVE,BATCH)
```

The /DISABLE qualifier is not followed by a keyword. Therefore, the qualifier disables the tracking of all resources. The /ENABLE qualifier then enables the tracking of the resources used by interactive and batch processes.

7.4. Starting Up a New Accounting File

To start up a new current accounting file, use the following command:

```
$ SET ACCOUNTING/NEW_FILE
```

This closes the current accounting file and opens a new version of it.

If the system encounters an error when trying to write information to the current accounting file, it automatically closes the file and opens a new version of it.

Example

This example closes the current accounting file, opens a new version of it, and changes the name of the old file to WEEK_24_RESOURCES.DAT. You can retain this file as a record of the resources used in that week.

```
$ SET ACCOUNTING/NEW_FILE
$ RENAME SYS$MANAGER:ACCOUNTNG.DAT;-1 WEEK_24_RESOURCES.DAT
```

7.5. Moving the Accounting File

When you first install your system, the current accounting file is SYS $MANAGER:ACCOUNTNG.DAT.

This file can become quite large. Moving it from your system disk can improve system performance.

How to Perform This Task

1. Define the logical name ACCOUNTNG in your system logical name table to point to the file you want to use. For example:

   ```
   $ DEFINE ACCOUNTNG MYDISK:[MYDIR]MYFILE.DAT/SYSTEM
   ```

   Give the full file specification, including the device and directory.

   Note

   Two nodes cannot log information in the same accounting file. If you define ACCOUNTNG on two nodes to point to the same file, each node will open and use its own version of the file.

   ---

2. To make the change permanent, add this definition to the file SYS $MANAGER:SYLOGICALS.COM.

3. Use the SET ACCOUNTING command with the /NEW_FILE qualifier to create and use the new file:

   ```
   $ SET ACCOUNTING/NEW_FILE
   ```

Example

This example changes the current accounting file to [MYDIR]MYDISK:MYFILE.DAT.

```
$ DEFINE ACCOUNTNG MYDISK:[MYDIR]MYFILE.DAT/SYSTEM
$ SET ACCOUNTING/NEW_FILE
```

7.6. Producing Reports of Resource Use

The three types of reports are:

Type of Report	Qualifier
Brief	/BRIEF (the default)
Full	/FULL
Summary	/SUMMARY

To produce a report, use the ACCOUNTING command with the appropriate qualifier in the following format:

```
ACCOUNTING [filespec[,...]/qualifier[,...]]
```

This runs the Accounting utility. The `filespec` parameter lists the accounting files you want to process. If you omit it, the Accounting utility processes the default current accounting file, SYS $MANAGER:ACCOUNTNG.DAT.

By default, the Accounting utility processes all the records in the accounting files you specify. You can use selection qualifiers to specify which records you want to process.

By default, brief and full reports present the records in the order in which they were logged in the accounting file. When you produce brief and full reports, you can use the /SORT qualifier to specify another order.

Example

This example produces a brief report of the information in the file that the logical name ACCOUNTNG points to. The /TYPE qualifier selects records for print jobs only. The /SORT qualifier displays them in reverse alphabetical order of user name.

```
$ ACCOUNTING ACCOUNTNG/TYPE=PRINT/SORT=-USER
    Date / Time      Type  Subtype  Username      ID     Source Status
-------------------------------------------------------------------------
13-APR-2016 13:36:04 PRINT          SYSTEM     20A00442         00000001
13-APR-2016 12:42:37 PRINT          JONES      20A00443         00000001
13-APR-2016 14:43:56 PRINT          FISH       20A00456         00000001
14-APR-2016 19:39:01 PRINT          FISH       20A00265         00000001
14-APR-2016 20:09:03 PRINT          EDWARDS    20A00127         00000001
14-APR-2016 20:34:45 PRINT          DARNELL    20A00121         00000001
14-APR-2016 11:23:34 PRINT          CLARK      20A0032E         00040001
14-APR-2016 16:43:16 PRINT          BIRD       20A00070         00040001
14-APR-2016 09:30:21 PRINT          ANDERS     20A00530         00040001
```

7.7. Setting Up Accounting Groups

Users are already organized into UIC security groups. For accounting purposes, security groups are often inappropriate. You can put users into accounting groups with the Authorize utility using the /ACCOUNT qualifier. In this way, each user is in an accounting group and a security group.

Using the Accounting utility, you can:

- Summarize the resources used by all the users in a particular accounting or security group. To do this, use the ACCOUNT or UIC keyword with the /SUMMARY qualifier.
- Select records for all the users in a particular accounting or security group. To do this, use the /ACCOUNT or /UIC qualifier.

How to Perform This Task

1. Plan your accounting groups. Decide which users you want in each accounting group, and choose names for the groups.

   The name of an accounting group can be a maximum of eight characters.

2. Change the account field values in the UAF. Use the Authorize utility's MODIFY command in the following format to change the value in the account field to the name of the user's accounting group:

   ```
   MODIFY username/ACCOUNT=account-name
   ```

   where:

`username`	is the name of the user
`account-name`	is the name of the accounting group that you want that user to be in

The next time your users log in, they will be in their new accounting groups, and their resource use will be tagged with the appropriate accounting group names.

Example

This example modifies the accounting group name to SALES_W8for the username FORD:

```
$ RUN SYS$SYSTEM:AUTHORIZE
UAF> MODIFY FORD/ACCOUNT=SALES_W8
UAF> EXIT
```

7.8. Monitoring Disk Space

To find out how much disk space a user is using, use SYSMAN or, if you have not enabled disk quotas, the DIRECTORY command.

How to Perform This Task

Use either of the following methods:

- Use the SYSMAN command DISKQUOTA SHOW in the following format:

  ```
  DISK QUOTA SHOW owner [/DEVICE=device-spec]
  ```

  This shows the number of blocks used by all the files that are owned by the specified user on the specified disk.

- Use the DIRECTORY command with the /SIZE and /GRAND_TOTAL qualifiers in the following format:

```
DIRECTORY [filespec[,...]]/SIZE=ALLOCATION/GRAND_TOTAL
```

This shows the number of blocks used by all the files in the specified file location.

Note that the DIRECTORY command does not include the blocks used by file headers or the user's root directory.

Examples

1. This example uses SYSMAN to find out the number of blocks used by all the files that are owned by each user.

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> DISKQUOTA SHOW *
%SYSMAN-I-QUOTA, disk quota statistics on device SYS$SYSTEM:MYDISK
Node UNION
     UIC                  Usage        Permanent Quota   Overdraft Limit
[0,0]                     0            1000              100
[DOC,EDWARDS]             115354       150000            5000
[DOC,FISH]                177988       250000            5000
[DOC,SMITH]               140051       175000            5000
[DOC,JONES]               263056       300000            5000
```

2. This example uses the DIRECTORY command to show the number of blocks allocated by all the files in and under MYDISK:[PARSONS].

```
$ DIRECTORY MYDISK:[PARSONS...]/SIZE=ALLOCATION/GRAND_TOTAL
Grand total of 28 directories, 2546 files, 113565 blocks.
```

Chapter 8. OpenVMS Cluster Considerations

This chapter describes concepts related to the OpenVMS Cluster environment; it also tells how the Show Cluster utility (SHOW CLUSTER) can display information about a cluster and how the System Management utility (SYSMAN) can help you manage an OpenVMS Cluster environment.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Beginning to use SHOW CLUSTER commands	*Section 8.3.2, "Beginning to Use SHOW CLUSTER Commands"*
Adding information to a report	*Section 8.3.3, "Adding Information to a Report"*
Controlling the display of data	*Section 8.3.4, "Controlling the Display of Data"*
Formatting the display of data	*Section 8.3.5, "Formatting the Display of Data"*
Creating a startup initialization file	*Section 8.3.6, "Creating a Startup Initialization File"*
Using command procedures containing SHOW CLUSTER commands	*Section 8.3.7, "Using Command Procedures Containing SHOW CLUSTER Commands"*
Using SYSMAN to manage security	*Section 8.5, "Using SYSMAN to Manage Security"*
Using the SYSMAN DO command to manage an OpenVMS Cluster	*Section 8.6, "Using the SYSMAN Command DO to Manage an OpenVMS Cluster"*

This chapter explains the following concepts:

Concept	Section
OpenVMS Cluster systems	*Section 8.1, "Understanding OpenVMS Cluster Systems"*
Setting up an OpenVMS Cluster environment	*Section 8.1.1, "Setting Up an OpenVMS Cluster Environment"*
Clusterwide system management	*Section 8.1.2, "Clusterwide System Management"*
The Show Cluster utility (SHOW CLUSTER)	*Section 8.3.1, "Understanding SHOW CLUSTER"*
SYSMAN and OpenVMS Cluster management	*Section 8.4, "Understanding SYSMAN and OpenVMS Cluster Management"*

8.1. Understanding OpenVMS Cluster Systems

An *VSI OpenVMS Cluster Systems Manual* is a loosely coupled configuration of two or more computers and storage subsystems, including at least one Alpha computer. An OpenVMS Cluster system appears as a single system to the user even though it shares some or all of the system resources. When a group of computers shares resources clusterwide, the storage and computing resources of all of the computers are combined, which can increase the processing capability, communications, and availability of your computing system.

A **shared resource** is a resource (such as a disk) that can be accessed and used by any node in an OpenVMS Cluster system. Data files, application programs, and printers are just a few items that can be accessed by users on a cluster with shared resources, without regard to the particular node on which the files or program or printer might physically reside.

When disks are set up as shared resources in an OpenVMS Cluster environment, users have the same environment (password, privileges, access to default login disks, and so on) regardless of the node that is used for logging in. You can realize a more efficient use of mass storage with shared disks, because the information about any device can be used by more than one node – the information does not have to be rewritten in many places. You can use the OpenVMS MSCP, which is the mass storage control protocol, or TMSCP, which is the tape mass storage control protocol, server software to make tapes accessible to nodes that are not directly connected to the storage devices.

You can also set up print and batch queues as shared resources. In an OpenVMS Cluster configuration with shared print and batch queues, a single queue database manages the queues for all nodes. The queue database makes the queues available from any node. For example, suppose your cluster configuration has fully shared resources and includes nodes ALBANY, BASEL, and CAIRO. A user logged in to node ALBANY can send a file that physically resides on node BASEL to a printer that is physically connected to node CAIRO, and the user never has to specify (or even know) the nodes for either the file or the printer.

Planning an OpenVMS Cluster System

A number of types of OpenVMS Cluster configurations are possible. Refer to *Guidelines for OpenVMS Cluster Configurations* and the OpenVMS Cluster Software Product Description (SPD) for complete information about supported devices and configurations.

The following sections briefly describe OpenVMS Cluster systems. For complete information about setting up and using an OpenVMS Cluster environment, refer to *VSI OpenVMS Cluster Systems Manual*.

8.1.1. Setting Up an OpenVMS Cluster Environment

Once you have planned your configuration, installed the necessary hardware, and checked hardware devices for proper operation, you can set up an OpenVMS Cluster system using various system software facilities. Setup procedures to build your cluster follow.

Procedure	**For More Information**
Installing or upgrading the operating system on the first OpenVMS Cluster computer	Installation and operations guide for your computer
Installing required software licenses	*VSI OpenVMS License Management Utility Guide*
Configuring and starting the DECnet for OpenVMS network	*VSI OpenVMS DECnet Networking Manual*
Configuring and starting TCP/IP Services	*VSI TCP/IP Services for OpenVMS Installation and Configuration*
Preparing files that define the cluster operating environment and that control disk and queue operations	*VSI OpenVMS Cluster Systems Manual*
Adding computers to the cluster	*VSI OpenVMS Cluster Systems Manual*

Depending on various factors, the order in which these operations are performed can vary from site to site, as well as from cluster to cluster at the same site.

8.1.2. Clusterwide System Management

Once any system is installed, you must decide how to manage users and resources for maximum productivity and efficiency while maintaining the necessary security. OpenVMS Cluster systems provide the flexibility to distribute users and resources to suit the needs of the environment. OpenVMS Cluster system resources can also be easily redistributed as needs change. Even with the vast number of resources available, you can manage the cluster configuration as a single system.

You have several tools and products to help you manage your cluster as a unified entity.

OpenVMS Cluster Tools

The following utilities are provided with the operating system:

Utility	**Description**
VSI DECamds	Collects and analyzes data from multiple nodes simultaneously, directing all output to a centralized DECwindows display. (Refer to *Section 8.2, "Using VSI DECamds to Analyze Data"* and the *DECamds User's Guide*.)

Utility	Description
Monitor utility (MONITOR)	Provides basic performance data. (See *Section 6.9, "Monitoring Operating System Performance"*.)
Show Cluster utility (SHOW CLUSTER)	Monitors activity in an OpenVMS Cluster configuration, and then collects and sends information about that activity to a terminal or other output device. (Described in *Section 8.3, "Using SHOW CLUSTER"*.)
System Management utility (SYSMAN)	Allows you to send common control commands across all, or a subset of, the nodes in the cluster. (Described in *Section 8.6, "Using the SYSMAN Command DO to Manage an OpenVMS Cluster"*.)

System Management Applications

The following products are *not* provided with the operating system:

Product	Description
POLYCENTER solutions	A comprehensive set of operations management products and services to help you manage complex distributed environments. However, the POLYCENTER Software Installation utility is described in this manual in *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.
[dag]Storage Library System (SLS) for VAX[ddag]Storage Library System (SLS) for Alpha	A set of software tools that enables tape, cartridge tape, and optical disks.
OpenVMS Cluster Console System (VCS)	Designed to consolidate the console management of the OpenVMS Cluster system at a single console terminal.

[dag]VAX specific
[ddag]Alpha specific

You can find additional information about these system management tools in the appropriate product documentation.

8.2. Using VSI DECamds to Analyze Data

VSI DECamds is areal-time monitoring, diagnostic, and correction tool that assists system managers to improve OpenVMS system and OpenVMS Cluster availability. DECamds can help system programmers and analysts target a specific node or process for detailed analysis, and can help system operators and service technicians resolve hardware and software problems.

DECamds simultaneously collects and analyzes system data and process data from multiple nodes and displays the output on a DECwindows Motif display. Based on the collected data, DECamds analyzes, detects, and proposes actions to correct resource and denial issues in real-time.

For more information, refer to the *DECamds User's Guide*.

8.3. Using SHOW CLUSTER

The Show Cluster utility (SHOW CLUSTER) monitors nodes in an OpenVMS Cluster system. You can use the utility to display information about cluster activity and performance.

The following sections describe the Show Cluster utility and explain how to perform these tasks:

Task	Section
Begin to use SHOW CLUSTER commands	*Section 8.3.2, "Beginning to Use SHOW CLUSTER Commands"*
Add information to a report	*Section 8.3.3, "Adding Information to a Report"*
Control the display of data	*Section 8.3.4, "Controlling the Display of Data"*
Format the display of data	*Section 8.3.5, "Formatting the Display of Data"*
Create a startup initialization file	*Section 8.3.6, "Creating a Startup Initialization File"*
Use command procedures containing SHOW CLUSTER commands	*Section 8.3.7, "Using Command Procedures Containing SHOW CLUSTER Commands"*

8.3.1. Understanding SHOW CLUSTER

You can display SHOW CLUSTER information on your terminal screen or send it to a device or a file. You can use SHOW CLUSTER interactively, with command procedures, or with an initialization file in which you define default settings. Because this utility is installed with the CMKRNL privilege, SHOW CLUSTER requires no special privilege.

SHOW CLUSTER information includes approximately 100 fields of data. You can customize the appearance of SHOW CLUSTER reports or define reports for access to often-needed data.

SHOW CLUSTER reports are organized by classes and fields:

Unit of Organization	Description
Class	Group of related fields of information. You can use class names to selectively add or remove an entire class from a report. Each class displays certain fields by default. Some classes have additional fields that you can add or remove using the field name.
Field	Column of data in a report. Use a unique field name to refer to each field of data. You can use the field name to selectively add or remove a field from reports. For the names and descriptions of all of the fields in each class, see the ADD (Field) command in the *VSI OpenVMS System Management Utilities Reference Manual*.

You can add fields or classes to the default SHOW CLUSTER report. If you add a field or class to a report in a continuous display, SHOW CLUSTER automatically adds the new data to the display.

Figure 8.1, "SHOW CLUSTER Default Display" shows a sample default SHOW CLUSTER report. The default report has two classes of information: SYSTEMS and MEMBERS. Below each class name are columns of fields that are associated with each class of information.

Figure 8.1. SHOW CLUSTER Default Display

View of Cluster from system ID 65536 node: CLUB 31DEC1997 14:00:00

SYSTEMS		MEMBERS
NODE	SOFTWARE	STATUS
CLUB	VMS V7.2	MEMBER
HSJ400	HSJ V25J	
HSC900	HSC V860	
CHIP	VMS V7.1	MEMBER
DISK3	RFX V256	
DISK1	RFX V256	
SPREE	VMS V6.2	MEMBER
SPRITZ	VMS V7.1	MEMBER

ZK8998AG E

Table 8.1, "Fields in Default SHOW CLUSTER Report" briefly describes the fields shown in *Figure 8.1, "SHOW CLUSTER Default Display"*.

Table 8.1. Fields in Default SHOW CLUSTER Report

Field Name	Description
NODE	Node name of the remote system. Normally, the cluster manager sets the node name using the system parameter SCSNODE. The node name should be the same as the DECnet for OpenVMS node name.
SOFTWARE	Name and version of the operating system currently running on the remote system.
STATUS	Status of the node in the cluster. (MEMBER indicates that the system is participating in the cluster.)

Over time, you can determine the most valuable classes and fields of data for your SHOW CLUSTER reports; you can then create a startup initialization file that establishes your default report formats. You can also build command procedures to use while running SHOW CLUSTER interactively. In this way, you can quickly reformat the report to show the data that is relevant for your installation. Start up initialization files and command procedures are explained later in this chapter.

Because SHOW CLUSTER information includes many fields of data, the report can quickly extend beyond screen limits. Therefore, SHOW CLUSTER provides mechanisms to help you control the display of data, including the following mechanisms:

- 38 SHOW CLUSTER commands
- A default keypad, which you can redefine

These mechanisms are described in detail in the *VSI OpenVMS System Management Utilities Reference Manual*.

8.3.2. Beginning to Use SHOW CLUSTER Commands

To use the Show Cluster utility, enter the SHOW CLUSTER command. If you specify the command without any qualifiers, however, SHOW CLUSTER simply displays a default report like that shown in *Figure 8.1, "SHOW CLUSTER Default Display"* and then displays the DCL prompt.

In a continuous display, on the other hand, you can enter SHOW CLUSTER commands to control report output. You can, for example, add classes or fields to, or remove classes or fields from, reports. To invoke a continuous display, in which you can enter SHOW CLUSTER commands, use the /CONTINUOUS qualifier on the SHOW CLUSTER command. (SHOW CLUSTER command qualifiers are described in *Section 8.3.2.3, "Using SHOW CLUSTER Qualifiers"*.)

How to Perform This Task

To invoke a continuous display of default SHOW CLUSTER report information, enter the following command:

```
$ SHOW CLUSTER/CONTINUOUS
```

SHOW CLUSTER then displays a default report. By default, SHOW CLUSTER updates the display every 15 seconds, with the changed data displayed in reverse video. After the default report, SHOW CLUSTER displays the following prompt:

```
Command>
```

(If the report extends below the limit of your terminal screen and you do not see the Command> prompt, you can press Return to display the prompt.)

The following sections contain instructions for performing beginning SHOW CLUSTER tasks:

Task	Section
Viewing information that is off the screen	*Section 8.3.2.1, "Viewing Information That Is Off the Screen"*
Exiting from a continuous display	*Section 8.3.2.2, "Exiting from a Continuous Display"*
Using SHOW CLUSTER qualifiers	*Section 8.3.2.3, "Using SHOW CLUSTER Qualifiers"*

8.3.2.1. Viewing Information That Is Off the Screen

The PAN command allows you to view the entire display by shifting your view of the display by column (horizontally) or by line (vertically).

Note

Report headings also move out of view as the reports in the display are panned beyond the limits of the screen. The SCROLL command, which is explained in *Section 8.3.5.4, "Scrolling a Report"*, preserves the headings as you scroll information. To use the SCROLL command, you must take the additional step of selecting a report if you have more than one report on the screen.

How to Perform This Task

To pan the display, perform one of the following actions:

- Enter PAN commands at the command prompt; for example:

  ```
  Command> PAN DOWN 10
  ```

  The command in this example moves the display down 10 lines.

- Define the arrow keys as PAN commands:

  ```
  Command> SET FUNCTION PAN
  ```

  This command redefines the arrow keys as follows:

Arrow Key	Redefinition
UP ARROW KEY	PAN UP 1
DOWN ARROW KEY	PAN DOWN 1
->	PAN RIGHT 1
<-	PAN LEFT 1

  You can then use the arrow keys to move up, down, right, and left in the display.

  Refer to the SET FUNCTION and PAN commands in the *VSI OpenVMS System Management Utilities Reference Manual* for details.

Resetting Arrow Keys

By default, the SHOW CLUSTER arrow keys are set to the EDIT function. This means that, at the command prompt, you can perform command line editing that is similar to DCL line-mode editing. For example, the left arrow key moves the cursor to the left, and the up arrow key recalls the previous command. Refer to the *VSI OpenVMS User's Manual* for information about DCL line-mode editing.

When you use the SET FUNCTION command, you reset the function keys. After that, the arrow keys are redefined and DCL line-mode editing is disabled.

To reset the arrow keys, enter the following command:

```
Command> SET FUNCTION EDIT
```

8.3.2.2. Exiting from a Continuous Display

To exit from a continuous display, perform one of the following actions:

- To return to the DCL prompt, perform one of the following actions:

- Enter EXIT after the Command> prompt.
- Press **Ctrl/Z**.
- Press **Ctrl/Y**.

- To exit without erasing the screen, press **Ctrl/C**.

8.3.2.3. Using SHOW CLUSTER Qualifiers

Table 8.2, "SHOW CLUSTER Qualifiers " briefly describes the qualifiers you can use with the SHOW CLUSTER command. The *VSI OpenVMS System Management Utilities Reference Manual* contains reference information about these SHOW CLUSTER qualifiers.

Table 8.2. SHOW CLUSTER Qualifiers

Qualifier	Function
/BEGINNING= `time`	Specifies the time that the SHOW CLUSTER session is to begin.
/CONTINUOUS	Controls whether SHOW CLUSTER runs as a continuously updating display.
/ENDING= `time`	Specifies the time that the SHOW CLUSTER session is to end.
/INTERVAL= `seconds`	Specifies the number of seconds that report information remains on the screen before it is updated.
/OUTPUT= `file-spec`	Directs the output from SHOW CLUSTER to the specified file instead of to the current SYS $OUTPUT device.

Example

In a continuous display, SHOW CLUSTER updates the display every 15 seconds by default. You can change this interval by using the /INTERVAL qualifier.

```
$ SHOW CLUSTER/CONTINUOUS/INTERVAL=5
```

In this example, SHOW CLUSTER updates reports every 5 seconds, displaying changed data in reverse video.

8.3.3. Adding Information to a Report

When you use the SHOW CLUSTER command, the resulting report is only part of the total information available. As shown in *Figure 8.1, "SHOW CLUSTER Default Display"*,the default classes displayed are MEMBERS and SYSTEMS. *Table 8.3, "Classes of Information Available in SHOW CLUSTER Reports"* briefly describes all the classes you can display in SHOW CLUSTER reports. Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for details about these classes.

Table 8.3. Classes of Information Available in SHOW CLUSTER Reports

Classes	Information Displayed
CIRCUITS	Describes virtual circuits on OpenVMS Cluster systems.

Classes	Information Displayed
CLUSTER	Shows general information about the OpenVMS Cluster system, such as the time it was formed, the last time a system joined or left, and the cluster quorum.
CONNECTIONS	Describes the connections established over a virtual circuit in the OpenVMS Cluster system
COUNTERS	Shows counts of the total accumulated traffic over a connection for the life of the connection.
CREDITS	Shows send and receive credit counts for connections in the OpenVMS Cluster system.
ERRORS	Displays a count of the errors on each port, along with information about the feasibility of reinitializing a port.
LOCAL_PORTS	Displays information about the local system interface to the OpenVMS Cluster system, such as the name, number, and status of each port, and the number of entries in the queues associated with each port.
MEMBERS	Describes systems actively participating in the OpenVMS Cluster system.
SYSTEMS	Describes all OpenVMS Cluster systems. It shows node name, identification number, hardware type, and software version.

Example

The following example shows how to add the CLUSTER class to a SHOW CLUSTER display:

```
Command> ADD CLUSTER
```

Figure 8.2, "SHOW CLUSTER Display with CLUSTER Report" shows the display that results from entering the ADD CLUSTER command. CLUSTER class is displayed below the default SHOW CLUSTER display.

Figure 8.2. SHOW CLUSTER Display with CLUSTER Report

View of Cluster from system ID 65536 node: CLUB 31DEC1997 14:00:00

SYSTEMS		MEMBERS
NODE	SOFTWARE	STATUS
CLUB	VMS V7.2	MEMBER
HSJ400	HSJ V25J	
HSC900	HSC V860	
CHIP	VMS V7.1	MEMBER
DISK3	RFX V256	
DISK1	RFX V256	
SPREE	VMS V6.2	MEMBER
SPRITZ	VMS V7.1	MEMBER

CLUSTER						
CL_EXP	CL_QUORUM	CL_VOTES	QF_VOTE	CL_MEMBERS	FORMED	LAST_TRANSITION
3	2	3	NO	4	15JUN1997	10DEC1997

ZK8999AGE

For descriptions of the fields in the CLUSTER class, refer to the SHOW CLUSTER section of the *VSI OpenVMS System Management Utilities Reference Manual*.

8.3.4. Controlling the Display of Data

Using SHOW CLUSTER commands, you can remove fields or classes from a display, remove broadcast messages from the screen, and refresh the screen display at any time. The following sections explain how to perform these operations.

8.3.4.1. Entering Commands to Display Data

SHOW CLUSTER allows you to customize the display of data during a continuous session by entering various commands. The *VSI OpenVMS System Management Utilities Reference Manual* describes SHOW CLUSTER commands in detail.

Updating of the continuous display stops as soon as you enter input from the terminal keyboard. When you press the Return key after entering a command, updating of the display resumes until you enter another command.

By default, updating takes place at 15-second intervals. If you do not enter a new command within 15 seconds, the command prompt disappears, and two more lines of data take its place.

8.3.4.2. Removing Broadcast Messages

When you receive a system broadcast message during a continuous SHOW CLUSTER session, the message appears at the bottom of the screen. A multiline message fills as many lines of the screen as it needs.

How to Perform This Task

The last broadcast message you receive remains on the screen until you acknowledge it by entering input from the terminal in one of the following ways:

- Press the **Return** key.
- Refresh the screen by pressing **Ctrl/W**.
- Enter a command.

If you receive more than one broadcast message, SHOW CLUSTER waits until the next update interval to display the next message.

SHOW CLUSTER also displays error messages at the bottom of the screen. For an explanation of the error messages, refer to the *OpenVMS System Messages: Companion Guide for Help Message Users*.

8.3.4.3. Refreshing the Screen

Ordinarily, a continuous display is updated or refreshed according to the default or specified interval time. SHOW CLUSTER scans the software databases, extracts and stores data for each field, displays any new or changed data, and updates the time. On VSI and VSI-compatible terminals, reverse video highlights any changed data.

How to Perform This Task

You can refresh the screen at any time by one of the following methods:

- Modify the format of the display with the ADD, REMOVE, INITIALIZE, or SET command.
- Use the REFRESH command.
- Press **Ctrl/W**.

8.3.5. Formatting the Display of Data

Because SHOW CLUSTER allows you to include additional fields and classes, you can produce reports that overflow the physical limits of the terminal screen. However, you can use a number of methods to modify the display to meet your needs:

Formatting Method	For More Information
Remove data from reports	*Section 8.3.5.1, "Removing Information from a Report"*
Modify field and screen size	*Section 8.3.5.2, "Modifying Field and Screen Size"*
Move a report	*Section 8.3.5.3, "Moving a Report"*
Scroll a report	*Section 8.3.5.4, "Scrolling a Report"*

8.3.5.1. Removing Information from a Report

You can remove certain fields or classes to reduce the width of are port to fit the limits of your screen. Also, certain fields or classes might not be important for your particular needs. You can also remove particular types of data to reduce the length of the report.

How to Perform This Task

Use the REMOVE command to remove fields or entire classes. To remove a field or class, use the appropriate qualifier with the REMOVE command. Refer to the REMOVE commands in the SHOW CLUSTER section of the *VSI OpenVMS System Management Utilities Reference Manual* for appropriate class names and qualifiers.

Examples

1. `Command> REMOVE SOFTWARE`

   The command in this example removes the SOFTWARE field from the SHOW CLUSTER report shown in *Figure 8.1, "SHOW CLUSTER Default Display"*.

   Refer to the ADD (Field) command description in the *VSI OpenVMS System Management Utilities Reference Manual* for a list of valid field names.

2. `Command> REMOVE MEMBERS`

   The command in this example removes the MEMBERS class from the SHOW CLUSTER report shown in *Figure 8.1, "SHOW CLUSTER Default Display"*.

8.3.5.2. Modifying Field and Screen Size

To make a report fit the physical limits of the screen, you can change the width of certain fields in the report. For example, if SHOW CLUSTER provides a field width that can contain any possible value and the values your cluster generates do not require that much space, you can adjust the field width with the SET (Field) command.

SHOW CLUSTER also allows you to adjust the size of the terminal screen. If the terminal is VSI-compatible and supports a wide report, you can set the screen to a width of up to 511 columns by specifying an appropriate value to the SET SCREEN command.

Examples

1. `Command> SET TRANSITION_TYPE/WIDTH=10`

   The command in this example sets the width of the TRANSITION_TYPE field to 10, which removes the time of day from the field but leaves the date.

2. `Command> SET SCREEN=132`

   The command in this example sets the screen width to 132.

Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for more details about using the SET (Field) and SET SCREEN commands.

8.3.5.3. Moving a Report

By default, SHOW CLUSTER operates with AUTO_POSITIONING ON. This means that the utility automatically arranges the reports to take best advantage of the available display space. However, you can position reports manually with the MOVE command, which implicitly sets AUTO_POSITIONING to OFF.

If you have multiple reports in your display, you must first select the report to be repositioned. You use the SELECT *window-name* command to specify the report name; for example:

- SCS (the default report, which usually includes fields in the SYSTEMS and MEMBERS classes)
- CLUSTER
- LOCAL_PORTS

Note

To select any report except the default SCS report, you must first add the class to the display if it is not already displayed; for example:

```
Command> ADD LOCAL_PORTS
```

As an alternative, you can repeatedly press the Select function key or the period key on the keypad to cycle from one report to the next. The selected report appears highlighted.

How to Perform This Task

To move a report, perform either of the following actions:

- Enter MOVE commands at the command prompt.

- Use the arrow keys that you define as MOVE commands.

  ```
  Command> SET FUNCTION MOVE
  ```

  This command redefines the arrow keys as follows:

Arrow Key	Redefinition
UP ARROW KEY	MOVE UP 1
DOWN ARROW KEY	MOVE DOWN 1
->	MOVE RIGHT 1
<-	MOVE LEFT 1

  When you enter a MOVE command, the display changes position by column (horizontally) or by line (vertically). For example, entering the command MOVE LEFT 5 moves the display 5 columns to the left. An empty frame appears around the new position of the report.

  When you are satisfied with the position of the report, enter the DESELECT command, which moves the report to the new position. Entering another SELECT command before the previous MOVE operation has been deselected also moves the report to its new position.

Example

```
Command> SELECT CLUSTER
Command> MOVE RIGHT 10
Command> DESELECT
```

The following lists explains the commands in the example:

1. The SELECT command selects the CLUSTER report (which is then highlighted).
2. The MOVE command positions the report frame 10 spaces to the right.
3. The DESELECT command terminates the MOVE operation and displays the contents of the report.

For more information, refer to the SELECT, SET FUNCTION, and DESELECT commands in the *VSI OpenVMS System Management Utilities Reference Manual*.

To reset the arrow keys, enter the following command:

```
Command> SET FUNCTION EDIT
```

8.3.5.4. Scrolling a Report

The SCROLL command provides a means of quickly scanning through a report without losing column headings. Scrolling scans a display by field (horizontally) and by line (vertically). The report headings remain stationary when you scroll vertically.

When the display has more than one report, you must first select a report by entering the SELECT command. The selected report is highlighted.

How to Perform This Task

To scroll a display, perform either of the following actions:

- Enter SCROLL commands at the command prompt.

- Use the arrow keys that you define as SCROLL commands.

  ```
  Command> SET FUNCTION SCROLL
  ```

  This command redefines the arrow keys as follows:

Arrow Key	Redefinition
UP ARROW KEY	SCROLL UP 1
DOWN ARROW KEY	SCROLL DOWN 1
->	SCROLL RIGHT 1
<-	SCROLL LEFT 1

Example

```
Command> SELECT SCS
Command> SET FUNCTION SCROLL
```

The commands in this example first select the SCS report (which is then highlighted), and then set the arrow keys to scroll functions. Refer to the SET FUNCTION and SCROLL commands in the *VSI OpenVMS System Management Utilities Reference Manual* for more information.

To reset the arrow keys, enter the following command:

```
Command> SET FUNCTION EDIT
```

8.3.6. Creating a Startup Initialization File

To customize the SHOW CLUSTER display, you can create a startup initialization file, which the utility executes when you enter it. SHOWCLUSTER takes the original default display, and adds or removes whatever classes or fields you specify. The resulting display becomes your default startup format. A startup initialization file resembles the following example:

```
!
!Startup Initialization File
!
!
INITIALIZE
REMOVE MEMBERS
ADD RP_REVISION,RP_TYPE,SYS_ID
SET SCREEN=132
```

This startup procedure deletes the MEMBERS class information from the default display. The procedure also adds the RP_REVISION and RP_TYPE fields from the CIRCUITS class and the SYS_ID field from the SYSTEMS class. The last line of the procedure sets the screen size to 132 columns.

How to Perform This Task

To create an initialization file, follow these steps:

1. Define the logical name SHOW_CLUSTER$INIT as `device:[directory]`SHCINI before invoking SHOW CLUSTER.

   For a startup file to execute before the display begins, you must assign the logical name SHOW_CLUSTER$INIT to the initialization file; for example:

```
DEFINE SHOW_CLUSTER$INIT DEVA:[JONES]SHCINI
```

When invoked, SHOW CLUSTER searches for the file defined by SHOW_CLUSTER$INIT. In this example, SHOW CLUSTER looks for DEVA:[JONES]SHCINI.INI when it starts up. If the initialization file is found, SHOW CLUSTER executes the procedure before beginning the display.

If you do not define SHOW_CLUSTER$INIT or it does not include a directory specification, SHOW CLUSTER searches the current default directory for a file named SHOW_CLUSTER.INI.

2. Customize the display using SHOW CLUSTER commands during a continuous SHOW CLUSTER session.

3. Preserve the command sequence by entering the following command:

```
Command> SAVE SHOW_CLUSTER$INIT.INI
```

You must specify SHOW_CLUSTER$INIT.INI, because the SAVE command creates a file with a file type of .COM by default. SHOW CLUSTER looks for an .INI file when it searches for a startup initialization file.

You can edit the file that the SAVE command creates to include comments or to improve its efficiency. For more information, refer to the SAVE command in the *VSI OpenVMS System Management Utilities Reference Manual*.

Instead of having SHOW CLUSTER build an initialization file, you can build one yourself in the same way you build a command procedure. The next section provides guidelines for creating a command procedure.

8.3.7. Using Command Procedures Containing SHOW CLUSTER Commands

You can create command procedures that contain SHOW CLUSTER commands. Such files let you modify display characteristics without having to enter commands interactively. You can use command procedures during a continuous SHOW CLUSTER session to perform a series of commands, for example, to customize the output of the display.

The following list contains guidelines for writing command procedures that contain SHOW CLUSTER commands:

- Use any valid SHOW CLUSTER commands.
- Nest command procedures up to 16 levels deep.
- Include the SHOW CLUSTER command INITIALIZE as the first command in the file. The INITIALIZE command ensures that the report is in a known state before any commands are executed to modify it.

Note

Do not include an EXIT command at the end of the command procedure. The EXIT command terminates SHOW CLUSTER and erases the SHOW CLUSTER display before you can see it.

Also, do not run SHOW CLUSTER command procedures from a batch job.

The following command procedure customizes a report display:

```
!
! Include only the node field from the default display; show votes
! and quorum for each node and for the cluster as a whole.
!
INITIALIZE
REMOVE SOFTWARE,STATUS
ADD VOTES,QUORUM,CL_VOTES,CL_QUORUM
```

This command procedure removes the SOFTWARE and STATUS fields from the report and adds fields that provide information about the cluster quorum and votes.

To execute a command procedure during a continuous SHOW CLUSTER session, specify the execute procedure (@) command, along with the file name of the command procedure. The default file type for command procedure files is .COM.

Example

The following command executes a command procedure named SYSMOD.COM:

```
Command> @SYSMOD
```

In this example, the default file type .COM is assumed because the file type is omitted.

For more information about creating command procedures, refer to the SAVE command in the *VSI OpenVMS System Management Utilities Reference Manual*.

8.4. Understanding SYSMAN and OpenVMS Cluster Management

The System Management utility (SYSMAN) provides two kinds of support for OpenVMS Cluster management:

- Cluster-specific commands, CONFIGURATION SET and CONFIGURATION SHOW, that you can use to manage security data and system time in a cluster
- Access to DCL-level commands with the DO command, which gives you the ability to apply a single DCL command across an entire cluster, rather than having to enter the command on each node

Each SYSMAN command requires a specific level of privilege. For more information about each command, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

8.5. Using SYSMAN to Manage Security

You can manage security data for an OpenVMS Cluster system with SYSMAN CONFIGURATION commands. *Table 8.4, "SYSMAN CONFIGURATION Commands"* summarizes these CONFIGURATION commands and their functions.

Table 8.4. SYSMAN CONFIGURATION Commands

Command	Function
CONFIGURATION SET CLUSTER_AUTHORIZATION	Modifies the group number and password in a local area cluster

Command	Function
CONFIGURATION SHOW CLUSTER_AUTHORIZATION	Displays the group number and multicast address of a local area cluster

8.5.1. Modifying the Group Number and Password

The group number identifies the group of nodes in the cluster, and the associated Ethernet address is used to send messages to all nodes in the cluster. The OpenVMS Cluster password protects the integrity of the cluster membership.

Using the CONFIGURATION SET CLUSTER_AUTHORIZATION command modifies the group number and password, as recorded in SYS$SYSTEM:CLUSTER_AUTHORIZE.DAT. Normally, you do not need to alter records in the CLUSTER_AUTHORIZE.DAT file.

If your configuration has multiple system disks, SYSMAN automatically updates each copy of CLUSTER_AUTHORIZE.DAT, provided that you have defined the environment as a cluster with the SET ENVIRONMENT/CLUSTER command.

Caution

If you change either the group number or password, you must reboot the entire cluster.

You cannot display the cluster password for security reasons, but you can display the group number and group multicast address with the CONFIGURATION SHOW CLUSTER_AUTHORIZATION command.

Examples

1. The following command example sets the environment to a specific cluster, sets privilege to SYSPRV, and modifies the cluster password:

```
SYSMAN> SET ENVIRONMENT/CLUSTER/NODE=NODE21
SYSMAN> SET PROFILE/PRIVILEGE=SYSPRV
SYSMAN> CONFIGURATION SET CLUSTER_AUTHORIZATION/PASSWORD=GILLIAN
%SYSMAN-I-CAFOLDGROUP, existing group will not be changed
%SYSMAN-I-GRPNOCHG, Group number not changed
SYSMAN-I-CAFREBOOT, cluster authorization file updated.
The entire cluster should be rebooted.
```

2. The following command example displays the group number and multicast address for NODE21. Because the group number and password on other nodes in the cluster are identical, no further information is displayed.

```
SYSMAN> CONFIGURATION SHOW CLUSTER_AUTHORIZATION
Node NODE21: Cluster group number 65240
Multicast address: AB-00-04-01-F2-FF
```

8.6. Using the SYSMAN Command DO to Manage an OpenVMS Cluster

The SYSMAN command DO enables you to execute a DCL command or command procedure on all nodes in the current environment. This is convenient when you are performing routine system management tasks on nodes in the OpenVMS Cluster system, such as:

- Installing images
- Starting up software
- Checking devices
- Checking memory

Each DO command executes as an independent process, so there is no process context retained between DO commands. For this reason, you must express all DCL commands in a single command string, and you cannot run a program that expects input.

In a cluster environment, SYSMAN executes the commands sequentially on all nodes in the cluster. Each command executes completely before SYSMAN sends it to the next node in the environment. Any node that is unable to execute the command returns an error message. SYSMAN displays an error message if the timeout period expires before the node responds.

In a dual-architecture heterogeneous OpenVMS Cluster running both OpenVMS VAX and OpenVMS Alpha, some uses of the DO command may require special handling. For example, if you are installing images that are named differently in each architecture, you can still use the DO command if you create logical name tables for VAX and for Alpha nodes. See the example sequence that follows this description for an example.

Some DCL commands, such as MOUNT/CLUSTER or SET QUORUM/CLUSTER, operate clusterwide by design. Similarly, operations on clusterwide logical names and tables operate clusterwide by design. It is best to avoid using these kinds of commands with the DO command in SYSMAN when the environment is set to cluster. As alternatives, you could leave SYSMAN temporarily with the SPAWN command and execute these commands in DCL, or you could define the environment to be a single node within the cluster.

Examples

1. The following example installs an image on a cluster. First, it adds CMKRNL and SYSPRV privileges to the current privileges because they are required by INSTALL and AUTHORIZE. The DO INSTALL command installs the file STATSHR. The DO MCR AUTHORIZE command sets up an account for user Jones, specifying a password and a default device and directory.

```
SYSMAN>SET PROFILE/PRIVILEGES=(CMKRNL,SYSPRV)/DEFAULT=SYS$SYSTEM
SYSMAN>DO INSTALL ADD/OPEN/SHARED WRKD$:[MAIN]STATSHR
SYSMAN>DO MCR AUTHORIZE ADD JONES/PASSWORD=COLUMBINE -
_SYSMAN>/DEVICE=WORK1/DIRECTORY=[JONES]
```

2. The following example sets the environment to cluster and starts up a software product called XYZ on each node in the cluster:

```
SYSMAN>SET ENVIRONMENT/CLUSTER
%SYSMAN-I-ENV, Current command environment:
        Clusterwide on local cluster
        Username SMITH    will be used on nonlocal nodes
SYSMAN> DO @SYS$STARTUP:XYZ_STARTUP
```

3. The following example shows how you can define logical names for VAX and Alpha nodes in a dual-architecture heterogeneous cluster, so that you can use the DO command to install architecture-specific images.

```
$ CREATE/NAME_TABLE/PARENT=LNM$SYSTEM_DIRECTORY SYSMAN$NODE_TABLE
```

```
$ DEFINE/TABLE=SYSMAN$NODE_TABLE ALPHA_NODES NODE21,NODE22,NODE23
$ DEFINE/TABLE=SYSMAN$NODE_TABLE VAX_NODES NODE24,NODE25,NODE26
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN>SET ENVIRONMENT/NODE=ALPHA_NODES
%SYSMAN-I-ENV, current command environment:
         Individual nodes: NODE21,NODE22,NODE23
         Username BOUCHARD will be used on nonlocal nodes

SYSMAN>DO INSTALL REPLACE SYS$LIBRARY:DCLTABLES.EXE
%SYSMAN-I-OUTPUT, command execution on node NODE21
%SYSMAN-I-OUTPUT, command execution on node NODE22
%SYSMAN-I-OUTPUT, command execution on node NODE23
SYSMAN> DO INSTALL REPLACE SYS$SYSTEM: DEC_FORTRAN.EXE
%SYSMAN-I-OUTPUT, command execution on node NODE21
%SYSMAN-I-OUTPUT, command execution on node NODE22
%SYSMAN-I-OUTPUT, command execution on node NODE23

SYSMAN>SET ENVIRONMENT/NODE=VAX_NODES
%SYSMAN-I-ENV, current command environment:
         Individual nodes: NODE24,NODE25,NODE26
         Username BOUCHARD will be used on nonlocal nodes

SYSMAN> DO INSTALL REPLACE SYS$LIBRARY:DCLTABLES.EXE
%SYSMAN-I-OUTPUT, command execution on node NODE24
%SYSMAN-I-OUTPUT, command execution on node NODE25
%SYSMAN-I-OUTPUT, command execution on node NODE26
SYSMAN> DO INSTALL REPLACE SYS$SYSTEM:FORTRAN$MAIN.EXE
%SYSMAN-I-OUTPUT, command execution on node NODE24
%SYSMAN-I-OUTPUT, command execution on node NODE25
%SYSMAN-I-OUTPUT, command execution on node NODE26
```

4. The following example shows which files are open on DISK2. You might use this if you want to dismount DISK2 and need to see which users in the cluster have files open.

```
SYSMAN>SET ENVIRONMENT/CLUSTER
%SYSMAN-I-ENV, Current command environment:
        Clusterwide on local cluster
        Username SMITH    will be used on nonlocal nodes
SYSMAN> DO SHOW DEVICE/FILES DISK2:

%SYSMAN-I-OUTPUT, command execution on node NODE21
Files accessed on device $1$DIA2: (DISK2, NODE22) on 14-MAY-2016
 15:44:06.05
Process name      PID     File name
                00000000  [000000]INDEXF.SYS;1
%SYSMAN-I-OUTPUT, command execution on node NODE22
Files accessed on device $1$DIA2: (DISK2, NODE21) on 14-MAY-2016
 15:44:26.93
Process name      PID     File name
                00000000  [000000]INDEXF.SYS;1
%SYSMAN-I-OUTPUT, command execution on node NODE23
Files accessed on device $1$DIA2: (NODE21, NODE22) on 14-MAY-2016
 15:45:01.43
Process name      PID     File name
                00000000  [000000]INDEXF.SYS;1
%SYSMAN-I-OUTPUT, command execution on node NODE24
Files accessed on device $1$DIA2: (NODE22, NODE21) on 14-MAY-2016
 15:44:31.30
```

```
Process name      PID     File name
                  00000000  [000000]INDEXF.SYS;1
Susan Scott       21400059  [SCOTT]DECW$SM.LOG;228
_FTA7:            214000DD  [SCOTT]CARE_SDML.TPU$JOURNAL;1
%SYSMAN-I-OUTPUT, command execution on node NODE25
Files accessed on device $1$DIA2: (NODE21, NODE22) on 14-MAY-2016
 15:44:35.50
Process name      PID     File name
                  00000000  [000000]INDEXF.SYS;1
DECW$SESSION      226000E6  [SNOW]DECW$SM.LOG;6
_FTA17:           2260009C  [SNOW.MAIL]MAIL.MAI;1
SNOW_1            2260012F  [SNOW.MAIL]MAIL.MAI;1
SNOW_2            22600142  [SNOW.MAIL]MAIL.MAI;1
SNOW_3            22600143  [SNOW.MAIL]MAIL.MAI;1
```

5. The following example shows how much memory is available on the nodes in a cluster. You might use this if you are installing software and want to know if each node has enough memory available.

```
SYSMAN> SET ENVIRONMENT/NODE=(NODE21,NODE22)
%SYSMAN-I-ENV, Current command environment:
        Clusterwide on local cluster
        Username SMITH    will be used on nonlocal nodes
SYSMAN> DO SHOW MEMORY
%SYSMAN-I-OUTPUT, command execution on node NODE21
              System Memory Resources on 14-MAY-2016 15:59:21.61
Physical Memory Usage (pages):     Total     Free    In Use  Modified
  Main Memory (64.00Mb)           131072    63955     65201      1916
Slot Usage (slots):                Total     Free  Resident   Swapped
  Process Entry Slots                360      296        64         0
  Balance Set Slots                  324      262        62         0
Fixed-Size Pool Areas (packets):   Total     Free    In Use      Size
  Small Packet (SRP) List          10568     1703      8865       128
  I/O Request Packet (IRP) List      375      925      2827       176
  Large Packet (LRP) List            157       28       129      1856
Dynamic Memory Usage (bytes):      Total     Free    In Use   Largest
  Nonpaged Dynamic Memory        1300480    97120   1203360     60112
  Paged Dynamic Memory           1524736   510496   1014240    505408
Paging File Usage (pages):                    Free  Reservable    Total
  DISK$MTWAIN_SYS:[SYS0.SYSEXE]SWAPFILE.SYS
                                             10000       10000    10000
  DISK$MTWAIN_SYS:[SYS0.SYSEXE]PAGEFILE.SYS
                                             60502      -52278   100000
Of the physical pages in use, 19018 pages are permanently allocated to
 VMS.

%SYSMAN-I-OUTPUT, command execution on node NODE22
              System Memory Resources on 14-MAY-2016 15:59:42.65
Physical Memory Usage (pages):     Total     Free    In Use  Modified
  Main Memory (32.00Mb)            65536    44409     20461       666
Slot Usage (slots):                Total     Free  Resident   Swapped
  Process Entry Slots                240      216        24         0
  Balance Set Slots                  212      190        22         0
Fixed-Size Pool Areas (packets):   Total     Free    In Use      Size
  Small Packet (SRP) List           5080     2610      2470       128
  I/O Request Packet (IRP) List     3101     1263      1838       176
  Large Packet (LRP) List             87       60        27      1856
Dynamic Memory Usage (bytes):      Total     Free    In Use   Largest
  Nonpaged Dynamic Memory        1165312   156256   1009056    114432
```

```
  Paged Dynamic Memory           1068032   357424    710608    352368
Paging File Usage (pages):                   Free  Reservable    Total
  DISK$MTWAIN_SYS:[SYS1.SYSEXE]SWAPFILE.SYS

                                            10000       10000    10000
  DISK$MTWAIN_SYS:[SYS1.SYSEXE]PAGEFILE.SYS
                                            110591      68443    120000
Of the physical pages in use, 9056 pages are permanently allocated to
 VMS.
```

Chapter 9. Network Considerations

This chapter discusses the following subjects:

- TCP/IP and DECnet networking software options for OpenVMS systems
- How to decide which networking software options are appropriate for your OpenVMS system
- How to prepare your system to join a network
- Where to obtain detailed information to help you install, configure, and manage the networking software you choose

The material provided in this chapter is intended as an introduction only. For complete planning, installation, configuration, use, and management information about the networking products, refer to the applicable product documentation.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Choosing the networking software appropriate for your system	*Section 9.2, "Choosing VSI Networking Software"*
Preparing to join a TCP/IP network	*Section 9.4, "Preparing to Join a TCP/IP Network"*
Installing and configuring TCP/IP Services	*Section 9.5, "Installing and Configuring TCP/IP Services"*
Starting and stopping TCP/IP Services	*Section 9.6, "Starting and Stopping TCP/IP Services"*
Preparing to join a DECnet-Plus network	*Section 9.9, "Preparing to Join a DECnet-Plus Network"*
Installing and configuring DECnet-Plus	*Section 9.10, "Installing and Configuring DECnet-Plus"*
Moving from DECnet Phase IV to DECnet-Plus	*Section 9.12, "Moving Your DECnet Phase IV*

Task	Section
	Network to DECnet-Plus"
Starting and stopping DECnet-Plus	*Section 9.13, "Starting and Stopping DECnet-Plus"*
Running DECnet applications over an IP network backbone	*Section 9.11, "Using DECnet Over TCP/IP"*

This chapter explains the following concepts:

Concept	Section
Networking software options for OpenVMS systems	*Section 9.1, "OpenVMS Networking Software Options"*
Introduction to VSI TCP/IP Services for OpenVMS software	*Section 9.3, "Understanding VSI TCP/IP Services for OpenVMS"*
Introduction to DECnet-Plus for OpenVMS software	*Section 9.8, "Understanding DECnet-Plus for OpenVMS Networking Software"*
Using DECnet over TCP/IP	*Section 9.11, "Using DECnet Over TCP/IP"*

9.1. OpenVMS Networking Software Options

The ability to connect your OpenVMS system with other systems and networks is a fundamental part of the OpenVMS operating system.

The following networking software options are available for OpenVMS systems:

- VSI TCP/IP Services for OpenVMS
- TCP/IP implementations for OpenVMS from vendors other than VSI
- VSI DECnet-Plus for OpenVMS (Phase V)
- VSI DECnet for OpenVMS (Phase IV)

You can install and use one of these networking options on an OpenVMS system, or you can install and use combinations of the options to accomplish the following:

- Interoperate with and share resources with other OpenVMS systems

- Interoperate with and share resources with systems running other operating systems, such as UNIX and Windows NT
- Use DECnet protocols and applications in combination with TCP/IP protocols and applications

9.2. Choosing VSI Networking Software

Table 9.1, "Choosing VSI Networking Software for OpenVMS Systems" describes the VSI layered networking software for OpenVMS systems. The software is available for you to install during the OpenVMS operating system installation or upgrade procedure. Or, you can install the software separately as layered products.

Table 9.1. Choosing VSI Networking Software for OpenVMS Systems

Product	Description
VSI TCP/IP Services for OpenVMS	TCP/IP Services is the VSI implementation of the industry-standard Transmission Control Protocol and Internet Protocol (TCP/IP) suite of protocols and Internet services for OpenVMS VAX and Alpha systems. If your OpenVMS system needs to communicate with heterogeneous networks (such as the Internet, UNIX systems, and Windows NT systems), you should choose VSI TCP/IP Services for OpenVMS software. With TCP/IP Services, you can connect to remote hosts and access files, exchange messages, develop applications, monitor the network, and perform other important tasks. You can use TCP/IP Services on your system as your only networking software. OpenVMS does not require that you use DECnet software. TCP/IP protocols can coexist with DECnet-Plus protocols. You can install TCP/IP Services to serve as your network backbone and install DECnet-Plus as well, so that you can continue to use DECnet applications and functionality. For information about using DECnet protocols over an IP backbone, see *Section 9.11, "Using DECnet Over TCP/IP"*. For an introduction to installing, using, and managing TCP/IP Services software, see *Section 9.3, "Understanding VSI TCP/IP Services for OpenVMS"*.
DECnet-Plus for OpenVMS (Phase V)	DECnet-Plus is the VSI Phase V implementation of the Digital Network Architecture (DNA). The software provides full backward compatibility with the older DECnet Phase IV product as well as the ability to run DECnet (NSP) and OSI over a DECnet, OSI, or TCP/IP network backbone. For information about using DECnet and OSI protocols over an IP backbone, see *Section 9.11, "Using DECnet Over TCP/IP"*. If your OpenVMS system needs to communicate using DECnet (Phase IV and Phase V) or OSI applications and protocols, and if your system needs to coexist with TCP/IP or OSI protocols, or both, you should choose DECnet-Plus software. If you are gradually upgrading your network from DECnet Phase IV, the DECnet-Plus features help you by allowing you to use Phase IV functionality as well as the newer OSI features and advantages.

Product	Description
	For an introduction to installing, using, and managing DECnet-Plus software, see *Section 9.8, "Understanding DECnet-Plus for OpenVMS Networking Software"*. **Note** Because DECnet-Plus contains both DNA Phase IV and Phase V protocols, you cannot run the separate DECnet Phase IV software product on the same system on which you install DECnet-Plus. For information about upgrading to DECnet-Plus, see the *VSI DECnet-Plus Planning Guide*.
DECnet for OpenVMS (Phase IV)	DECnet Phase IV is the VSI Phase IV implementation of the Digital Network Architecture (DNA). The product does not include the OSI protocols and the TCP/IP communications capability of the later DECnet-Plus (Phase V) product. You can run TCP/IP Services software on your system along with DECnet Phase IV, but you cannot run IP as the network backbone. If you intend to use your OpenVMS system to communicate in the traditional OpenVMS environment rather than a heterogeneous environment, you might want to choose DECnet Phase IV, the older DECnet product. This chapter does not contain summary product information about the older DECnet Phase IV product. For information about using DECnet Phase IV, refer to the *VSI OpenVMS DECnet Networking Manual* and the *OpenVMS Performance Management Manual* manual (available on the OpenVMS Documentation CD-ROM).

Table 9.2, "VSI Networking Software Interoperability Options" lists possible networking software combinations for communication between OpenVMS systems.

Table 9.2. VSI Networking Software Interoperability Options

If System A has...	And System B has...	Systems A and B can communicate using...
TCP/IP Services	TCP/IP Services	TCP/IP applications
DECnet Phase IV	DECnet Phase IV	DECnet applications
DECnet-Plus	DECnet-Plus	DECnet applications OSI applications
DECnet-Plus	DECnet Phase IV	DECnet applications
DECnet-Plus	OSI	OSI applications
TCP/IP Services and DECnet Phase IV	TCP/IP Services	TCP/IP applications
TCP/IP Services and DECnet Phase IV	DECnet Phase IV	DECnet applications

If System A has...	And System B has...	Systems A and B can communicate using...
TCP/IP Services and DECnet-Plus	TCP/IP Services	TCP/IP applications
TCP/IP Services and DECnet-Plus	DECnet-Plus	DECnet applications OSI applications
TCP/IP Services and DECnet-Plus	TCP/IP Services and DECnet-Plus	OSI applications DECnet applications DECnet applications using DECnet over TCP/IP OSI applications using OSI over TCP/IP TCP/IP applications
TCP/IP Services and DECnet-Plus	OSI (supporting RFC 1006) and TCP/IP Services	OSI applications OSI over TCP/IP TCP/IP applications
TCP/IP Services and DECnet-Plus	OSI (not supporting RFC 1006) and TCP/IP Services	OSI applications TCP/IP applications

9.3. Understanding VSI TCP/IP Services for OpenVMS

The VSI TCP/IP Services for OpenVMS product is the OpenVMS implementation of the industry-standard TCP/IP suite of communications protocols as specified in Request for Comments (RFCs) used by the Internet Engineering Task Force (IETF).

With TCP/IP, heterogeneous networks can interconnect, making it possible for users to connect to remote hosts in many ways:

- Network file access. Users can access files on remote hosts.
- Electronic mail. Users can exchange messages between hosts.
- Application development. Application programmers can develop TCP/IP client/server applications for communication between local and remote hosts.
- Download and file transfer. Users can exchange files between hosts.
- User information. Users can access information about other users logged onto the local or remote host.
- Remote management. System managers can monitor the network and applications from remote hosts.

- Remote terminal access. Users can access a remote host as if their terminal is connected directly to that host.
- Remote command execution. Users can issue commands to remote hosts.
- Remote printing. Users can send or receive print jobs to or from remote printers.
- Remote file copy. Users can copy files that reside on remote hosts.
- Remote booting. Servers can provide boot information for remote clients.

Internet working with TCP/IP hides the hardware details of each individual network and allows computers to communicate independently of their physical network connections. TCP/IP provides both a standard transport mechanism and full-duplex, reliable, stream communication services for software applications.

VSI TCP/IP Services for OpenVMS software provides interoperability and resource sharing between OpenVMS systems, UNIX systems, and other systems that support the TCP/IP protocol suite and Sun Microsystems' Network File System (NFS). TCP/IP systems and other internet hosts share data and resources by using standard TCP/IP protocols over a number of network hardware configurations: Ethernet, Fiber Distributed Data Interface (FDDI), Token Ring, and asynchronous transfer mode (ATM).

Each end system connected to a TCP/IP network is called a **host**. Each host has a unique name and address. The local host is the system you are using, and the remote host is the system with which you are communicating. Hosts are connected by **lines** that carry information between the hosts. The line is the physical path over which data can pass from one host to another. (Examples of lines are telephone lines, fiber-optic cables, and satellites.)

A TCP/IP network is called a packet-switching network. Information is transmitted in small packets of data rather than as a continuous stream from host to host. For example, a file to be transmitted from one host to another is divided into many small packets that are sent across the network one at a time. Each packet contains information about the address of the destination host. At the destination, the packets are reassembled.

The process of directing a data message from a source host to a destination host is called **routing**. For hosts not directly connected to each other, data can be forwarded from the source to the destination through intervening hosts.

9.3.1. Support for OpenVMS Cluster Systems

VSI TCP/IP Services for OpenVMS supports OpenVMS Cluster systems and the use of cluster aliases. The network sees the cluster as one system with one name, called the internet alias. A remote host can use the cluster alias to address the cluster as one host or use the host name of a cluster member to address a cluster member individually.

9.3.2. TCP/IP Services Management Tools and Utilities

VSI TCP/IP Services for OpenVMS provides a comprehensive, easy-to-use network management tool that includes over 100 OpenVMS DCL-style commands. These commands allow you to locally configure, monitor, and tune TCP/IP Services components by issuing management commands at the `TCPIP>` prompt.

You can also use UNIX management commands to manage some components.

For detailed information about managing TCP/IP Services on your system, refer to the *VSI TCP/IP Services for OpenVMS Management* guide.

9.4. Preparing to Join a TCP/IP Network

Before you can configure a host to run TCP/IP, you must have a unique IP address and host name. Unlike network hardware addresses, which are hardcoded and fixed, an IP address is assigned by a network administrator. Networks connected to the public Internet must obtain an official, unique network ID from the Inter NIC. The IP address has a total of 32 bits (four octets) that identify the network and host. A host name is the name assigned to a computer to facilitate communication.

In addition to providing host and network IDs, the IP address provides information that helps a host determine which packets it should receive and which packets it should ignore. In order for a host to know whether or not another host is on the same or a different subnet, the host compares its own address and the address of the destination computer to a subnet mask. If the network ID of the destination matches the network ID of the source, the packet is delivered to the destination host on the local network. If the network IDs do not match, the packet is forwarded through an IP router to the destination computer.

In addition to an IP address and host name, you must obtain the following information from your network administrator:

- The address for the default gateway, if your systems needs to communicate with TCP/IP hosts not on the local network
- The routing protocol the network uses
- The address of the domain name server (or servers) that will resolve host names into IP addresses
- The network subnet and broadcast masks

You must also decide which end-user services to provide and whether your system is operating as a client, server, or both.

For more information about planning to install and configure VSI TCP/IP Services for OpenVMS, refer to *VSI TCP/IP Services for OpenVMS Concepts and Planning*.

9.5. Installing and Configuring TCP/IP Services

You can install VSI TCP/IP Services for OpenVMS software on your system in either of the following ways:

- From the OpenVMS installation procedure menu, as part of the operating system upgrade or installation
- As a layered application using the DCL command PRODUCT INSTALL

Once TCP/IP Services software is installed successfully, you invoke the menu-based configuration procedure, as follows, to configure the software according to the unique characteristics of your system and network:

```
$ @SYS$MANAGER:TCPIP$CONFIG
```

For detailed installation and configuration information, refer to the *VSI TCP/IP Services for OpenVMS Installation and Configuration* manual.

9.6. Starting and Stopping TCP/IP Services

To start VSI TCP/IP Services for OpenVMS software on your system after you configure the standard software and optional components, or to stop the software for an orderly shutdown, enter:

```
$ @SYS$MANAGER:TCPIP$CONFIG
```

Then enter the appropriate menu option to start or stop TCP/IP Services.

9.7. TCP/IP Services Documentation

Table 9.3, "VSI TCP/IP Services for OpenVMS Documentation" lists the documentation that supports VSI TCP/IP Services for OpenVMS. For detailed information about how to plan for, install, and use TCP/IP Services, refer to these documents.

Table 9.3. VSI TCP/IP Services for OpenVMS Documentation

Manual	Contents
VSI TCP/IP Services for OpenVMS Release Notes	Describes changes to the software including installation, upgrade, and compatibility information. These notes also describe new and existing software problems and restrictions, and software and documentation corrections.
VSI TCP/IP Services for OpenVMS Concepts and Planning	Introduces TCP/IP concepts and components and provides information to help you plan your software configuration.
VSI TCP/IP Services for OpenVMS Installation and Configuration	Explains how to install and configure the VSI TCP/IP Services for OpenVMS product on you OpenVMS host.
VSI TCP/IP Services for OpenVMS User's Guide	Describes how to use the applications available with TCP/IP Services such as remote file operations, E-mail, TELNET, TN3270, and network printing. Also explains how to use these services to communicate with systems on private internets or on the worldwide Internet.
VSI TCP/IP Services for OpenVMS Management	Describes day-to-day management of the VSI TCP/IP Services for OpenVMS software product.
VSI TCP/IP Services for OpenVMS Management Command Reference	Describes the VSI TCP/IP Services for OpenVMS management commands. A companion guide to the *VSI TCP/IP Services for OpenVMS Management* manual.
VSI TCP/IP Services for OpenVMS Tuning and Troubleshooting	Describes how to troubleshoot VSI TCP/IP Services for OpenVMS and how to tune the performance of the product.
VSI TCP/IP Services for OpenVMS Guide to IPv6	Describes the VSI TCP/IP Services for OpenVMS IPv6 features and how to install and configure IPv6 on your system.

Manual	Contents
VSI TCP/IP Services for OpenVMS Sockets API and System Services Programming	Describes how to use VSI TCP/IP Services for OpenVMS to develop network applications using Berkeley Sockets or OpenVMS system services.
VSI TCP/IP Services for OpenVMS ONC RPC Programming	Provides an overview of high-level programming with open network computing remote calls (ONC RPC), describes how to use the RPCGEN protocol compiler to create applications, and describes the RPC programming interface.
VSI TCP/IP Services for OpenVMS SNMP Programming and Reference	Describes the Simple Network Management Protocol (SNMP) and the SNMP application programming interface (eSNMP). It describes the subagents provided with TCP/IP Services, utilities provided for managing subagents, and how to build you own subagents.

9.8. Understanding DECnet-Plus for OpenVMS Networking Software

DECnet-Plus for OpenVMS provides the means for various VSI operating systems to communicate with each other and with systems provided by other vendors. The DECnet-Plus network supports remote system communication, resource sharing, and distributed processing. Network users can access resources on any system in the network. Each system participating in the network is known as a network **node**.

DECnet-Plus is the implementation of the fifth phase of the Digital Network Architecture (DNA). DNA Phase V integrates the OSI protocols with DECnet protocols. In addition, DECnet-Plus includes support for the Internet standard RFC 1006 and the Internet draft RFC 1859, allowing OSI and DECnet applications to run over TCP/IP. Thus, using DECnet-Plus, applications can communicate with peer OSI and DECnet applications on any DECnet Phase IV-based system or OSI-based system, whether from VSI or from other vendors.

The support of the non-proprietary protocols is the primary difference between DECnet (Phase IV) and DECnet-Plus.

DECnet-Plus provides many features designed to enhance networking capabilities. These features include:

- Global name/directory services that allow large networks to easily store, manage, and access addressing information for (potentially) millions of network objects, such as end systems, users, printers, files, and disks.
- Optional local name/directory services for smaller networks that do not have as critical a need for global directory services.
- Expanded network management capabilities with the Network Control Language (NCL).
- Host-based routing that allows an OpenVMS system to operate as a DECnet-Plus intermediate system in a routing domain.

  This feature is especially useful for those configurations where you need to route from a LAN to a WAN and want to use an existing system to do the routing rather than investing in a dedicated router.

Host-based routing is not intended for use in network configurations that have high-throughput requirements.

- Increased addressing capacity with the OSI standard address format, making possible unique addressing for virtually an unlimited number of nodes. Existing Phase IV addresses can continue to be used for systems upgraded to DECnet-Plus. The Phase IV address is automatically converted by the configuration procedure to the OSI address format (as a DECnet Phase IV compatible address).
- Address auto configuration that enables an adjacent router to configure the node address for the local node.
- Single configuration for OSI components.

  X.25 (on VAX systems),wide area device drivers (WANDD), File Transfer, Access, and Management (FTAM), and Virtual Terminal (VT) applications are included with DECnet-Plus software. (On Alpha systems, X.25 support is separate from DECnet-Plus software.)

9.8.1. DECnet-Plus Node Names

Naming conventions for DECnet node names correspond to the two types of DECnet functionality:

- DECnet-Plus full names

  Full names are hierarchically structured DECnet node names that can be stored in a DECdns name service. Full names can be a maximum of 255 bytes long.
- DECnet Phase IV node names, called node synonyms in DECnet-Plus

  These names are the shorter names used by DECnet Phase IV, restricted to six or fewer characters. Using these names enables DECnet-Plus to be backward compatible with DECnet Phase IV systems in the same network.

Syntax for Full Names

Full names have the following general syntax:

```
namespace:.directory ...  .directory.node-name
```

where:

namespace	Identifies the global name service
directory ... .directory	Defines the hierarchical directory path within the name service
node-name	Is the specific object defining the DECnet node

The following examples show node full names for the Local name space, DECdns, and DNS/BIND, respectively:

```
Local namespace -  LOCAL:.CPlace
DECdns          -  ACME:.warren.CPlace
Domain          -  CPlace.warren.acme.com
```

The system stores a full name as you enter it, preserving uppercase and lowercase entries. However, when matching an entry with a stored full name, the system is case insensitive; in other words, if the user enters Acme, the system recognizes it as equivalent to ACME.

For more information about full names, refer to the DECnet-Plus documentation.

9.8.2. Support for OpenVMS Cluster Systems

DECnet-Plus software supports OpenVMS Cluster systems and the use of cluster aliases. DECnet-Plus allows for three aliases for each OpenVMS Cluster. DECnet Phase IV nodes cannot be DECnet-Plus alias members. (A separate alias must be configured for use with DECnet Phase IV nodes.)

The CLUSTER_CONFIG.COM command procedure performs an OpenVMS Cluster configuration. It can configure all members of a cluster from any cluster member. It invokes the DECnet-Plus for OpenVMS NET$CONFIGURE.COM command procedure to perform any required modifications to NCL initialization scripts. Use CLUSTER_CONFIG.COM to configure an OpenVMS Cluster. Use NET$CONFIGURE.COM directly to configure additional DECnet-Plus satellite nodes once CLUSTER_CONFIG.COM has already been used.

9.8.3. DECnet-Plus Management Tools and Utilities

DECnet-Plus for OpenVMS provides a variety of network tools that let you perform the following tasks:

- Manage local and remote DECnet Phase V components. Two interfaces are available:the Network Control Language (NCL) command-line interface and a Motif-based windows interface (NET $MGMT.)

  VSI supplies the DECNET_MIGRATE tool that converts individual DECnet Phase IV NCP commands to NCL commands, and NCP commands within command procedures to NCL commands. You can use DECNET_MIGRATE when you are new to DECnet-Plus, learning NCL, and want help specifying familiar NCP commands in NCL syntax.

- Manage remote DECnet Phase IV nodes with the NCP Emulator (NCP.EXE). This utility supports a significant range of NCP commands. It is not designed to replace NCL for managing a DECnet-Plus system.

- Use DECnet-Plus for OpenVMS initialization scripts (files in the form SYS$MANAGER:NET $*.NCL).

- Perform maintenance operations (using MOP) such as downline load, up line dump, remote console connection, and loopback testing support. DECnet-Plus for OpenVMS provides enhanced support and performance for concurrent downline loads. For more information about MOP and how to start this process, refer to the *VSI DECnet-Plus for OpenVMS Network Management Guide* guide.

- Perform enhanced event logging using EVD.

- Use Common Trace Facility (CTF) for troubleshooting.

- Use the DECNET_REGISTER tool to assist in managing node names in your network for the Local and DECdns name spaces.

9.9. Preparing to Join a DECnet-Plus Network

Before configuring your DECnet-Plus node, you must make some decisions regarding addressing, the use of name services, time services, and routers. You must also be aware of license dependencies unique to X.25 software.

For detailed planning information, refer to the *VSI DECnet-Plus Planning Guide*.

9.10. Installing and Configuring DECnet-Plus

You can install DECnet-Plus for OpenVMS software on your system in either of the following ways:

- From the OpenVMS installation procedure menu, as part of the operating system upgrade or installation
- As a layered application using the DCL command PRODUCT INSTALL

Once the DECnet-Plus software is installed successfully, you invoke a menu-based configuration procedure to configure the software according to the unique characteristics of your system and network:

- Use the Fast configuration option if you are upgrading from a DECnet Phase IV node, you plan to use the existing Phase IV configuration, and your node is not part of an OpenVMS Cluster. Enter:

  ```
  $ @SYS$MANAGER:NET$CONFIGURE
  ```

- Use the Basic configuration option if your node is in a cluster, you are upgrading or reconfiguring DECnet-Plus, and you want to run DECnet over TCP/IP. Enter:

  ```
  $ @SYS$MANAGER:NET$CONFIGURE BASIC
  ```

- Use the Advanced configuration option if your node's configuration is complex and you need to customize it. Enter:

  ```
  $ @SYS$MANAGER:NET$CONFIGURE ADVANCED
  ```

For detailed installation and configuration information, refer to the *DECnet-Plus for OpenVMS Installation and Basic Configuration Manual* manual.

9.11. Using DECnet Over TCP/IP

The DECnet over TCP/IP feature extends the DECnet Phase V architecture to coexist and communicate with TCP/IP networks. This feature requires a valid DECnet license and a licensed and installed TCP/IP product that supports the PATHWORKS Internet Protocol (PWIP) interface.

You should consider enabling DECnet over TCP/IP for the following purposes:

- Transition from DECnet to TCP/IP. To move to TCP/IP-based networks gradually based on business need. DECnet over TCP/IP functionality is designed to be transparent to users.
- Coexistence. To run DECnet or OSI applications, or both, in a TCP/IP-based network.

With DECnet over TCP/IP, you can:

- Expand your DECnet network to communicate over TCP/IP network backbones. DECnet over TCP/IP allows combinations of naming services (DECdns, large LOCAL file, and DNS/BIND).
- Expand the network by joining two existing DECnet networks without the need to renumber the nodes.
- Use IP-only traffic in part or all of the backbone. With DECnet over TCP/IP, DECnet or OSI applications can run over the lower levels of the TCP/IP protocol stack and the IP network backbone.
- Enable the feature on a node-to-node basis or throughout the entire network.

The DECnet over TCP/IP functionality uses TCP/IP to form a logical link between DECnet nodes. It uses the PATHWORKS IP Driver to interface with TCP. DECnet applications run transparently between DECnet nodes connected by TCP/IP. Users on those DECnet nodes can connect to each other using DECnet node synonyms or IP full names. For example:

```
$ SET HOST SYSABC
$ SET HOST SYSABC.boston.acme.com
$ SET HOST 16.12.42.19
```

To enable DECnet over TCP/IP on your system, you must do the following primary tasks:

- Install and configure DECnet-Plus on the system.

  When you run the NET$CONFIGURE procedure, specify the Domain directory service (for TCP/IP addresses) as well as the Local or DNS/BIND service when prompted by the configuration procedure.

- Install and configure TCP/IP Services on the system.

  When you run the TCPIP$CONFIG configuration procedure, enable the PWIP driver to form a bridge between DECnet-Plus and TCP/IP Services software. The PWIP driver is listed as Option 1 on the Optional Components menu.

For detailed information about enabling and using DECnet over TCP/IP, refer to the *DECnet-Plus for OpenVMS Applications Installation and Advanced Configuration Guide* and *VSI DECnet-Plus for OpenVMS Network Management Guide* manuals.

9.12. Moving Your DECnet Phase IV Network to DECnet-Plus

If you are planning to transition your network from DECnet Phase IV to DECnet-Plus, you can move portions of the network to DECnet-Plus or move the entire network. Because DECnet-Plus is backward compatible, you can choose to run your system and the network in the same manner as you have before, using DECnet Phase IV applications, routing, and so forth. You can implement the additional functionality available to you from DECnet-Plus any time you are ready. The changes mostly involve network management. They are almost entirely transparent to users and applications.

A number of automated tools (DECnet transition utilities and the NCP Emulator) are available, in addition to the simplified configuration procedures, to help ease the transition to full DECnet-Plus functionality.

For detailed information about transitioning your network, refer to the *VSI DECnet-Plus Planning Guide* manual.

9.13. Starting and Stopping DECnet-Plus

If you install DECnet-Plus from the OpenVMS installation menu, the software starts automatically. If you need to restart DECnet-Plus for any reason (for example, after shutting down the network by executing SYS$STARTUP:NET$SHUTDOWN.COM), enter the following command:

```
$ @SYS$STARTUP:NET$STARTUP
```

To shut down DECnet-Plus software, which disables and deletes various network entities on your system, enter:

```
$ @SYS$MANAGER:NET$SHUTDOWN
```

9.14. DECnet-Plus Documentation

Table 9.4, "DECnet-Plus for OpenVMS Documentation" lists the documentation that supports the DECnet-Plus for OpenVMS software. For complete information about how to plan for, install, configure, and manage DECnet-Plus, refer to these documents.

Table 9.4. DECnet-Plus for OpenVMS Documentation

Manual	Contents
DECnet-Plus for OpenVMS Release Notes	Describes changes to the software; installation, upgrade, and compatibility information; new and existing software problems and restrictions; and software and documentation corrections. You can print this text file from the configuration procedure.
VSI DECnet-Plus for OpenVMS Introduction and User's Guide	Provides an introduction to networking on the system and includes user information.
DECnet-Plus for OpenVMS Installation and Basic Configuration Manual	Explains how to install DECnet-Plus and perform the BASIC configuration option.
DECnet-Plus for OpenVMS Applications Installation and Advanced Configuration Guide	Explains how to install and configure network applications and perform the ADVANCED configuration procedure option.
DECnet-Plus for OpenVMS Installation and Quick Reference	Provides a quick reference for upgrading your system to DECnet-Plus during installation.
VSI DECnet-Plus Planning Guide	Provides steps for transitioning from DECnet Phase IV functionality to DECnet Phase V.
DECnet-Plus for OpenVMS Network Management and Quick Reference Card	Provides a quick reference for DECnet-Plus network management.
VSI DECnet-Plus for OpenVMS Network Management Guide	Includes network management concepts and tasks for DECnet-Plus systems.
VSI DECnet-Plus for OpenVMS Network Control Language Reference Guide	Provides command descriptions and examples for all NCL commands.
VSI DECnet-Plus for OpenVMS Problem Solving Guide	Explains how to isolate and solve DECnet-Plus problems that can occur while the network is running, and how to perform loopback tests.

Chapter 10. Managing the Local Area Network (LAN) Software

This chapter describes how the LAN software works and the tasks you must perform to manage the LAN software on your system.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Running the LANACP LAN server process	*Section 10.3.1, "Running the LANACP LAN Server Process"*
Invoking and running LANCP	*Section 10.4.1, "Invoking and Running LANCP"*
Managing LAN devices	*Section 10.5, "Managing LAN Devices"*
Managing the LAN device databases	*Section 10.6, "Managing the LAN Device Databases"*
Managing the LAN node databases	*Section 10.7, "Managing the LAN Node Databases"*
Migrating from DECnet MOP to LAN MOP	*Section 10.8.2, "Migrating from DECnet MOP to LAN MOP"*
Using CLUSTER_CONFIG_LAN.COM and LAN MOP	*Section 10.8.3, "Using CLUSTER_CONFIG_LAN.COM and LAN MOP"*
Managing the MOP downline load services	*Section 10.9, "Managing the LAN MOP Downline Load Service"*
Initiating the MOP console carrier	*Section 10.9.8, "MOP Console Carrier"*

Task	Section
Requesting MOP trigger boot	*Section 10.9.9, "MOP Trigger Boot"*
Using LAN Failover	*Section 10.10, "Understanding LAN Failover"*

This chapter explains the following concepts:

Concept	Section
Local area networks	*Section 10.1, "Understanding Local Area Networks"*
LANACP LAN server process	*Section 10.3, "Understanding the LANACP LAN Server Process"*
LANCP utility	*Section 10.4, "Understanding the LANCP Utility"*
MOP downline load services	*Section 10.8, "Understanding LAN MOP"*
LAN Failover	*Section 10.10, "Understanding LAN Failover"*

10.1. Understanding Local Area Networks

A local area network (LAN) provides a communications channel designed to connect information processing equipment in a limited area such as a room, a building, or a complex of buildings (for example, a campus). Nodes in a LAN can be linked by the following types of data transmission media:

- Ethernet—One of the earliest and the most common LANs. Ethernet can refer to either a general LAN application (for example, Ethernet address) or to the specific CSMA/CD (carrier sense multiple access with collision detection) technology that implements the Intel, Xerox, and VSI inter company Ethernet specifications.

  There are three types of Ethernet LANs as follows:

  - Ethernet (802.3) with transmission speeds of 10Mbps
  - Fast Ethernet (802.3u) with transmission speeds of 100Mbps
  - Gigabit Ethernet (802.3z) with transmission speeds of 1000Mbps

  All three types of Ethernet employ CSMA/CD protocol, and the same frame format and same frame size.

- FDDI (Fiber Distributed Data Interface) — Implemented as dual-ring, token ring LANs.

- Token Ring—The IEEE 802.5 standard token passing ring.
- ATM (Asynchronous Transfer Mode) — The following standard is supported:
  - LAN emulation over ATM supports the ATM Forum's LAN Emulation V1.0 (LANE) standard.
  - Classical IP over ATM supports the RFC 1577 standard. (DGLTA, DGLPA, DGLPB only)

10.1.1. LAN Characteristics

LAN controllers are devices that, along with additional external hardware, implement the Ethernet, FDDI, Token Ring, LAN emulation over ATM or Classical IP (RFC 1577) specifications. A LAN controller and the local system constitute a node. The LAN controller communicates with the local system through the system bus, and with remote systems implementing the Ethernet, FDDI, Token Ring, or LAN emulation over ATM specifications through the communication medium. (The Ethernet specification is described in *The Ethernet–Data Link Layer and Physical Layer Specification*. The FDDI specifications are available from ANSI. The Token Ring specifications are available from IEEE. The LAN emulation over ATM specifications are available from the ATM Forum.)

Application programs use the LAN driver's QIO interface to perform I/O operations to and from other nodes on the LAN. For detailed information about the QIO interface, refer to the *VSI OpenVMS I/O User's Reference Manual*. *Table 10.1, "Characteristics of LAN Media"* provides a brief summary of the differences between the types of LAN media.

Table 10.1. Characteristics of LAN Media

Media	Speed	Maximum Frame Size	Maximum Cable Lengths [dag]
Ethernet 802.3	10Mbps	1518 bytes	100 meters
Fast Ethernet 802.3u	100Mbps	1518 bytes	100Base-TX – 100 meters 100Base-FX – 412 meters (one-half duplex) – 2016 meters (full-duplex)
Gigabit Ethernet 802.3z	1000Mbps	1518 or 9018 bytes	1000Base-SX fiber optic – 550 meters 1000Base-LX fiber optic – 5 kilometers 1000Base-CX copper shielded – 25 meters 1000BaseT copper UTP – 100 meters
FDDI	100Mbps	4495 bytes	40 kilometers
Token Ring 802.5	4 or 16 Mbps	4462 bytes	300 meters
LAN emulation over ATM	155Mbps or 622Mbps	1516, 4544, or 9234	2 kilometers, 300 meters

[dag] Larger networks can be constructed with hubs, bridges, and switches.

10.1.1.1. Ethernet LANs

An Ethernet is a cable to which each system or device is connected by a single line. In an office or other area where personal computers and workstations are located, ThinWire Ethernet or unshielded twisted-pair cabling is usually used.

Individual systems can either be connected directly to an Ethernet or gain access to an Ethernet by means of a local area interconnect device, such as a DELNI. A DELNI serves as a concentrator, grouping systems together. Many similar devices, such as hubs, repeaters, and switches also provide the connectivity.

10.1.1.2. FDDI LANs

FDDI uses a dual ring of trees topology. It uses one ring as the primary ring, the other ring as a backup, and the tree configuration for increased flexibility, manageability, and availability.

FDDI networks and Ethernet networks can be combined to form a single extended LAN. This lets applications running on a system connected to FDDI communicate with applications that run on a system connected to Ethernet.

An FDDI concentrator provides for the attachment of FDDI devices such as VAX and Alpha nodes or FDDI-Ethernet bridges to the FDDI LAN.

10.1.1.3. Token Ring LANs (Alpha Only)

Token Ring controllers use either shielded or unshielded twisted pairs of wire to access the ring. Note that it is difficult to connect a Token Ring LAN directly bridged to any other type of LAN. However, routing protocols to other LANs work easily.

10.1.1.4. ATM LANs (Alpha Only)

LANs over ATM consist of a connection-oriented network based on cell switching. The OpenVMS ATM network uses AAL5 ATM adaption layer for data transmission.

For LAN emulation over ATM, OpenVMS implements only the LAN emulation client (LEC) and does not implement the LAN emulation server (LES), the Broadcast and Unknown (BUS), or the LAN emulation Configuration Server (LECS). The LES, BUS, and LECS must be provided by some other facility such as the ATM switch. OpenVMS supports eight LAN emulation clients per ATM adapter.

Classical IP over ATM (CLIP) implements a data-link level device that has the same semantics as an Ethernet interface (802.3). This interface is used by a TCP/IP protocol to transmit 802.3 (IEEE Ethernet) frames over an ATM network. The model that OpenVMS follows for exchanging IP datagrams over ATM is based on RFC1577 (Classical IP over ATM).

10.1.2. LAN Addresses

Nodes on the LAN are identified by unique addresses. A message can be sent to one, several, or all nodes on the LAN simultaneously, depending on the address used.

Upon application, IEEE assigns a block of addresses to a producer of LAN nodes. Thus, every manufacturer has a unique set of addresses to use. Normally, one address out of the assigned block of physical addresses is permanently associated with each controller (usually in read-only memory).

This address is known as the hardware address of the controller. Each controller has a unique hardware address.

A LAN address is 48 bits in length. LAN addresses are represented as six pairs of hexadecimal digits (six bytes) separated by hyphens (for example, AA-01-23-45-67-FF). The bytes are displayed from left to right in the order in which they are transmitted; bits within each byte are transmitted from right to left. In this example, byte AA is transmitted first; byte FF is transmitted last.

A LAN address can be a physical address of a single node or a multicast address, depending on the value of the low-order bit of the first byte of the address (this bit is transmitted first). The two types of node addresses are:

- Physical address—The unique address of a single node on a LAN. The least significant bit of the first byte of a physical address is 0. (For example, in physical address AA-00-03-00-FC-00, byte AA in binary is 1010 1010, and the value of the low-order bit is 0.)
- Multicast address—A multidestination address of one or more nodes on a given LAN. The least significant bit of the first byte of a multicast address is 1. (For example, in the multicast address 0B-22-22-22-22-22, byte 0B in binary is 0000 1011, and the value of the low-order bit is 1.)

  Token Ring devices do not support IEEE 802 standard multicast addresses. They do support functional addresses. A functional address is a locally administered group address that has 31 possible values. Each functional address sets one bit in the third through sixth bytes of the address, and bytes 1 and 2 are 03-00 (C0:00 in bit reversed format). To convert a multicast address to a functional address, use the SET DEVICE/MAP command.

10.2. Managing Local Area Networks

The local area network (LAN) software includes two system management tools that work in conjunction with the OpenVMS LAN driver system software:

- Local Area Network Control Program (LANCP)
- LAN Auxiliary Control Program (LANACP) LAN server process

The LAN system management tools:

- Allow you to set LAN parameters to customize your LAN environment.
- Display LAN settings and counters.
- Provide Maintenance Operations Protocol (MOP) downline load support for devices such as terminal servers, x-terminals, and LAN-based printers, and for booting satellites in an OpenVMS Cluster environment. This MOP support provides an alternative to the traditional method of using either DECnet for OpenVMS or DECnet/OSI software.

Table 10.2, "LAN System Management Enhancements" describes the LAN management software and the functionality supported on systems running OpenVMS Alpha and OpenVMS VAX.

Table 10.2. LAN System Management Enhancements

Utility	Description	OpenVMS Support
LAN Auxiliary Control Program (LANACP)	Runs as a server process whose primary function is to provide MOP downline load service. Other services	The LANACP utility provides identical functionality on VAX and Alpha systems running OpenVMS Version 7.0 and later.

<table>
<tr><th>Utility</th><th>Description</th><th>OpenVMS Support</th></tr>
<tr><td></td><td>include maintenance of a LAN volatile device database and a LAN volatile node database.</td><td></td></tr>
<tr><td>LAN Control Program (LANCP)</td><td>Allows you to control LAN software parameters and obtain information from the LAN software. You can use the LANCP utility to:
Obtain LAN device counters, revision, and configuration information
Change the operational parameters of LAN devices on the system
Maintain the permanent and volatile LAN device and node databases
Control the LANACP LAN server process (including MOP downline load server related functions)
Initiate MOP console carrier connections
Send MOP trigger boot requests to other nodes</td><td>OpenVMS Alpha Version 6.1 contained the initial implementation of LANCP, which did not include MOP-related functions.

OpenVMS Version 6.2 (VAX and Alpha) added MOP-related functions and extended some of this capability to VAX systems. The following table shows how the LAN utility functions are currently supported on VAX and Alpha systems:<table><tr><th>Function</th><th>OpenVMS Alpha Version 7.3-1</th><th>OpenVMS Alpha Version 7.3-1</th></tr><tr><td>Change operational parameters of LAN devices?</td><td>Yes</td><td>No</td></tr><tr><td>Display LAN device information?</td><td>Yes</td><td>Limited</td></tr><tr><td>Support MOP functions?</td><td>Yes</td><td>Yes</td></tr></table></td></tr>
</table>

10.3. Understanding the LANACP LAN Server Process

You can run the LANACP LAN server process to provide the following services:

- Maintenance of the LAN volatile node database
- Maintenance of the LAN volatile device database
- MOP downline load

The LANCP utility allows you to issue instructions to the LANACP process.

Three principal files are connected with LANACP:

- SYS$SYSTEM:LANACP.EXE

  This file is the LANACP utility program.

- SYS$STARTUP:LAN$STARTUP.COM

  This file starts the LANACP server process.

- SYS$STARTUP:VMS$DEVICE_STARTUP.COM

  This file contains an entry that is used to start LANACP automatically at system startup.

In addition, four system logical names, described in *Table 10.3, "LANACP System Logical Names"*, are associated with the LANACP LAN server process.

Table 10.3. LANACP System Logical Names

Component	Description
LAN$DLL	Defines the location of downline load files, where the location of the file is not provided in the load request or explicitly defined in the LAN volatile node database. By default, this is defined as SYS $SYSROOT:[MOM$SYSTEM].
LAN$NODE_DATABASE	Defines the location of the LAN permanent node database. By default, this is defined as SYS$COMMON:[SYSEXE]LAN $NODE_DATABASE.DAT.
LAN$DEVICE_DATABASE	Defines the location of the LAN permanent device database. By default, this is defined as SYS$SPECIFIC:[SYSEXE]LAN $DEVICE_DATABASE.DAT.
LAN$ACP	Defines the location of the LANACP LAN server process log file, containing entries describing changes to the LAN permanent device and node databases, and load request and load status information. By default, this is defined as SYS$MANAGER:LAN$ACP.LOG.

10.3.1. Running the LANACP LAN Server Process

To start the LANACP LAN server process, type @SYS$STARTUP:LAN$STARTUP at the DCL prompt.

10.3.2. Stopping the LANACP LAN Server Process

To stop the LANACP LAN server process, enter the SET ACP/STOP command at the LANCP utility prompt.

10.4. Understanding the LANCP Utility

The LANCP utility allows you to set and show LAN parameters. *Section 10.4.1, "Invoking and Running LANCP"* describes how to invoke the LANCP utility. *Table 10.4, "Functions of the LANCP Utility"* describes LAN functions and provides section references to the LANCP commands that help you perform these functions.

Table 10.4. Functions of the LANCP Utility

Task	Section
Managing LAN devices	*Section 10.5, "Managing LAN Devices"*

Task	Section
Managing LAN device databases	*Section 10.6, "Managing the LAN Device Databases"*
Managing LAN node databases	*Section 10.7, "Managing the LAN Node Databases"*
Managing the MOP downline load service	*Section 10.9, "Managing the LAN MOP Downline Load Service"*
Initiating a MOP console carrier connection	*Section 10.9.8, "MOP Console Carrier"*
Sending MOP trigger boot requests	*Section 10.9.9, "MOP Trigger Boot"*

10.4.1. Invoking and Running LANCP

Table 10.5, "Invoking the LANCP Utility" describes the ways you can invoke and run the LANCP utility (SYS$SYSTEM:LANCP.EXE).

Table 10.5. Invoking the LANCP Utility

Command	Example
Use the RUN command	At the DCL command prompt, enter: `$ RUN SYS$SYSTEM:LANCP` The LANCP utility displays the LANCP prompt at which you can enter LANCP commands.
Define LANCP as a foreign command	Either at the DCL prompt or in a startup or login command file, enter: `$ LANCP :== $SYS$SYSTEM:LANCP` Then, you can enter the command LANCP at the DCL prompt to invoke the utility and enter LANCP commands. When you enter the LANCP command: ● Without specifying any command qualifiers, the LANCP utility displays the LANCP prompt at which you can enter commands. ● With command qualifiers, the LANCP utility terminates after it executes the command and returns you to the DCL prompt.
Use the MCR command	At the DCL command prompt, enter: `$ MCR LANCP`

Command	Example
	When you enter the MCR LANCP command: • Without specifying any command qualifiers, the LANCP utility displays the LANCP prompt at which you can enter commands. • With command qualifiers, the LANCP utility terminates after it executes the command and returns you to the DCL prompt.

At the LANCP> prompt, you can enter LANCP commands.

For information about the LANCP utility, enter the HELP command at the LANCP> prompt.

To exit from the LANCP utility, enter the EXIT command or press **Ctrl/Z** at the LANCP> prompt.

10.4.2. LANCP Commands

Table 10.6, "LANCP Commands" summarizes the LANCP commands.

Table 10.6. LANCP Commands

Command	Function
@ (Execute Procedure)	Executes a command procedure.
CLEAR DEVICE	Deletes a device from the LAN volatile device database.
CLEAR DLL	Clears MOP downline load counters for all nodes and devices.
CLEAR MOPDLL	Same as the CLEAR DLL command.
CLEAR NODE	Deletes a node from the LAN volatile node database.
CONNECT NODE	Connects to a LAN device, such as a terminal server, that implements a management interface using the MOP console carrier protocol.
CONVERT DEVICE_DATABASE	Converts a device database to the current format required by LANCP.
CONVERT NODE_DATABASE	Converts a node database to the current format required by LANCP.
DEFINE DEVICE	Enters a device into the LAN permanent device database or modifies an existing entry.
DEFINE NODE	Enters a node into the LAN permanent node database or modifies an existing entry.
EXIT	Stops execution of LANCP and returns control to the DCL command level.
HELP	Provides online help information about the LANCP utility.
LIST DEVICE	Displays information in the LAN permanent device database.
LIST NODE	Displays information in the LAN permanent node database.
PURGE DEVICE	Deletes a device from the LAN permanent device database.
PURGE NODE	Deletes a node from the LAN permanent node database.
SET ACP	Modifies the operation of the LANACP LAN server process.
SET DEVICE	Modifies device characteristics in the LAN volatile device database and in the device itself.

Command	Function
SET NODE	Enters a node into the LAN volatile node database or modifies an existing entry.
SHOW CONFIGURATION	Displays a list of LAN devices on the system.
SHOW DEVICE	Displays information in the LAN volatile device database.
SHOW DLL	Displays the current state of MOP downline load services.
SHOW LOG	Displays recent downline load activity.
SHOW MOPDLL	Same as the SHOW DLL command.
SHOW NODE	Displays information in the LAN volatile node database.
SPAWN	Creates a subprocess of the current process.
TRIGGER NODE	Issues a request to reboot to a remote node.

For detailed information about LANCP commands and qualifiers, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

10.4.3. LANCP Miscellaneous Functions

Use the SPAWN command to create a subprocess of the current process. The SPAWN command copies the context of the subprocess from the current process. This allows you to exit temporarily from LANCP without having to restart LANCP when you resume.

The syntax for the SPAWN command is as follows:

```
SPAWN [command-string]
```

You can set up the LANCP utility to execute commands from a command file from within LANCP. The LANCP utility recognizes the command file as the file name preceded by the at sign (@). The default file name extension is .COM.

10.5. Managing LAN Devices

LAN device management consists of displaying device characteristics and setting device parameters. You can use the LANCP utility to set parameters for the types of LAN devices shown in *Table 10.7, "LAN Devices"*.

Table 10.7. LAN Devices

LAN	Device Examples	Description
Ethernet	DE425, DE434, DE435, DE436, DE500, DECchip 21040, DEMNA	Allow the selection of media type (type of cable connected) and the speed of connection (Ethernet or FastEthernet). Allow full-duplex operation (point-to-point operation between a similar device or between the device and a switch).
FDDI	DEFTA, DEFPA, DEFAA, DEFEA, DEMFA	Allow full-duplex operation.

LAN	Device Examples	Description
Token Ring [ddag]	DETRA, DW300, DW110, TC4048	Allow the setting of Token Ring parameters and the definition of source routing and functional address mapping.
All	Any	Allow the setting of generic parameters such as the number of receive buffers.
ATM [ddag]	DGLTA, DGLPA, DGLPB, DAPBA, DAPCA	Allow the setting of Emulated LAN (ELAN) parameters.

[ddag] Alpha only

10.5.1. Displaying System Devices

To display the LAN devices on the system, enter the SHOW CONFIGURATION command using the following syntax:

```
SHOW CONFIGURATION
```

Example

```
LANCP> SHOW CONFIGURATION

LAN Configuration:
    Device   Medium      Default LAN Address   Version
    ------   ------      -------------------   -------
     EWA0    CSMA/CD      08-00-2B-E4-00-BF    02000023
     EWB0    CSMA/CD      08-00-2B-92-A4-0D    02000023
     IRA0    Token Ring   00-00-93-58-5D-32    20000223
```

This command displays the output from a SHOW CONFIGURATION command entered on a node that has three LAN devices: two DE435s, and a DETRA:.

The version is the device-specific representation of the actual version. In this example, for two devices on the PCI bus, the actual version is in the low byte (2.3 for the DE435 adapters). A device that does not have a readable version is shown as version zero.

Consult your device-specific documentation to correlate the version returned with a particular hardware or firmware implementation of the device.

10.5.2. Displaying Device Characteristics

To display information about a LAN device (in the volatile device database), enter the LANCP command SHOW DEVICE using the following syntax:

```
SHOW DEVICE device-name [/qualifiers]
```

Table 10.8, "SHOW DEVICE Command Qualifiers" provides a brief description of the SHOW DEVICE command qualifiers.

Note

If you do not specify a qualifier, the utility displays the matching devices without additional information.

Table 10.8. SHOW DEVICE Command Qualifiers

Qualifier	Description
/ALL	Shows all devices which match device name.
/CHARACTERISTICS	Same as the /PARAMETERS qualifier.
/DLL	It shows downline load characteristics.
/COUNTERS	Displays device counters.
/INTERNAL_COUNTERS	Displays internal counters. By default, it does not display zero counters. To see all counters, including zero, use the additional qualifier /ZERO. To see debug counters, use the additional qualifier /DEBUG.
/MAP	Displays the current configuration of the functional address mapping table.
/MOPDLL	Same as the /DLL qualifier.
/OUTPUT[ddag]	Directs output to the specified filename.
/PARAMETERS	Displays status and related information about the device.
/REVISION	Displays the current firmware revision of the adapter, if available or applicable.
/SR_ENTRY	Displays the contents of the current source routing cache table.
/TRACE	Displays LAN driver trace data.

[ddag] Alpha only

Examples

1.
```
LANCP> SHOW DEVICE/COUNTERS EXA0

Device Counters EXA0:
              Value  Counter
              -----  -------
             259225  Seconds since last zeroed
            5890496  Data blocks received
            4801439  Multicast blocks received
             131074  Receive failure
          764348985  Bytes received
          543019961  Multicast bytes received
                  3  Data overrun
            1533610  Data blocks sent
             115568  Multicast packets transmitted
             122578  Blocks sent, multiple collisions
              86000  Blocks sent, single collision
             189039  Blocks sent, initially deferred
          198120720  Bytes sent
           13232578  Multicast bytes transmitted
            7274529  Send failure
                  0  Collision detect check failure
                  0  Unrecognized frame destination
                  0  System buffer unavailable
                  0  User buffer unavailable
```

   This SHOW DEVICE command displays counters for Ethernet device EXA0.

2.
```
LANCP> SHOW DEVICE/MAP ICA0
  Multicast to Functional Address Mapping ICA0:
    Multicast address   Functional Address   Bit-Reversed
```

```
-----------------   ------------------   ------------
09-00-2B-00-00-04   03-00-00-00-02-00    C0:00:00:00:40:00
09-00-2B-00-00-05   03-00-00-00-01-00    C0:00:00:00:80:00
CF-00-00-00-00-00   03-00-00-08-00-00    C0:00:00:10:00:00
AB-00-00-01-00-00   03-00-02-00-00-00    C0:00:40:00:00:00
AB-00-00-02-00-00   03-00-04-00-00-00    C0:00:20:00:00:00
AB-00-00-03-00-00   03-00-08-00-00-00    C0:00:10:00:00:00
09-00-2B-02-00-00   03-00-08-00-00-00    C0:00:10:00:00:00
09-00-2B-02-01-0A   03-00-08-00-00-00    C0:00:10:00:00:00
AB-00-00-04-00-00   03-00-10-00-00-00    C0:00:08:00:00:00
09-00-2B-02-01-0B   03-00-10-00-00-00    C0:00:08:00:00:00
09-00-2B-00-00-07   03-00-20-00-00-00    C0:00:04:00:00:00
09-00-2B-00-00-0F   03-00-40-00-00-00    C0:00:02:00:00:00
09-00-2B-02-01-04   03-00-80-00-00-00    C0:00:01:00:00:00
09-00-2B-02-01-07   03-00-00-02-00-00    C0:00:00:40:00:00
09-00-2B-04-00-00   03-00-00-04-00-00    C0:00:00:20:00:00
09-00-2B-02-01-00   03-00-00-00-08-00    C0:00:00:00:10:00
09-00-2B-02-01-01   03-00-00-00-10-00    C0:00:00:00:08:00
09-00-2B-02-01-02   03-00-00-00-20-00    C0:00:00:00:04:00
03-00-00-00-00-01   03-00-00-00-00-01    C0:00:00:00:00:80
03-00-02-00-00-00   03-00-02-00-00-00    C0:00:40:00:00:00
```

This SHOW DEVICE command displays mapping information for Token Ring device ICA0.

3.
```
LANCP> SHOW DEVICE/PARAM IRA0
 Device Parameters IRA0:
              Value  Parameter
              -----  ---------
             Normal  Controller mode
           External  Internal loopback mode  00-00-93-58-5D-32  Hardware
 LAN address
         Token Ring  Communication medium
            Enabled  Functional address mode
                 No  Full duplex enable
                 No  Full duplex operational
                 16  Line speed (megabits/second)
            16 Mbps  Ring speed
                STP  Line media
            Enabled  Early token release
           Disabled  Monitor contender
                200  SR cache entries
                  2  SR discovery timer
                 60  SR Aging Timer
            Enabled  Source routing
                  3  Authorized access priority  AA-00-04-00-92-FF
 Upstream neighbor
                  0  Ring number
```

This SHOW DEVICE command displays status and parameter information for Token Ring device IRA0.

4.
```
LANCP> SHOW DEVICE/REVISION FXA0
Device revision FXA0:  05140823
```

This command displays revision information for FDDI device FXA0.

5.
```
LANCP> SHOW DEVICE/SR_ENTRY ICA0
 Source Routing Cache Table ICA0:
```

```
   LAN address       State    XmtTmo   RcvTmo  StaleTmo DiscvTmo
-----------------   -----   -------- -------- -------- --------
AA-00-04-00-92-FF   LOCAL   00000028 00000028 00000245 00000000
```

This SHOW DEVICE command displays source routing entry information for Token Ring device ICA0.

10.5.3. Setting Device Characteristics

All LAN devices are characterized by a collection of parameters. The parameters define the operational characteristics of a LAN device on the medium to which the device is connected.

To set LAN device parameters directly, enter the SET DEVICE command at the LANCP> prompt. The LANCP utility issues this command to the LANACP server process, which then issues the appropriate QIOs to set the driver characteristics.

The syntax for the SET DEVICE command is:

```
SET DEVICE device-name [/qualifiers]
```

Table 10.9, "DEFINE DEVICE and SET DEVICE Command Qualifiers" provides a brief description of the SET DEVICE command qualifiers that apply directly to LAN devices.

Table 10.9. DEFINE DEVICE and SET DEVICE Command Qualifiers

Qualifier	Description
/AGING_TIMER= `value`	Sets the amount of time in seconds to age source routing cache entries before marking them stale.
/ALL	Sets data for all LAN devices.
/ATMADDRESS=LES [ddag]	Sets the LAN emulation server (LES) address for asynchronous transfer mode (ATM). Usually the address is not user specified, and this qualifier is used only if you want a specific address. By default the address is determined by software from the configuration server for the LES. The /ATMADDRESS=LES qualifier's syntax is as follows: `SET DEVICE/ATMADDRESS = ([NO]LES=the ATM server)`
/ATMADDRESS=ARP [ddag]	Sets the address resolution protocol (ARP) server address for Classical IP over ATM. This qualifier is required before a LIS is enabled if the local host is not the ARP server. The /ATMADDRESS=ARP qualifier's syntax is as follows: `SET DEVICE/ATMADDRESS = (ARP=atm_arp_server)`
/AUTONEGOTIATE (default) /NOAUTONEGOTIATE	Enables or disables the use of auto-negotiation to determine the link settings. You may need to disable link auto-negotiation when connected to a switch or device that does not support auto-negotiation.
/CACHE_ENTRIES= `value`	Sets the number of entries to reserve for caching source routing address entries.
/CLIP [ddag]	Sets up the Classical Internet Protocol (CLIP) over ATM (RFC1577). The CLIP qualifier implements a data-link level

Qualifier	Description

	device as a client and/or a server in a logical IP subnet (LIS). This allows the IP protocol to transmit Ethernet frames over the ATM network. The /CLIP = ENABLE command causes the system to join the LIS. The /CLIP = DISABLE command causes the client to leave the logical IP subnet.
	
	Note that a LIS requires a server, and there must be only one server for each subnet. Communication between subnets can only be performed by a router. There can only be one client for each ATM adapter.
	
	The /CLIP qualifier's syntax with standard Internet dotted notation is as follows:
	
	`SET DEVICE/CLIP =(ip_subnet=a.b.c.d,`
	`ip_address=a.b.c.d,`
	`parent=device,`
	`name="ip subnet name",`
	`enable, disable`
	`type = client|server)`
	
	The meanings of the syntax for /CLIP are as follows:
	
	● *ip_address*
	
	Specifies the IP address of the CLIP client.
	
	● *subnet_mask*
	
	Specifies the subnet mask of the CLIP client.
	
	● *parent*
	
	Parent device name.
	
	● *name*
	
	Specifies a name for the LIS to aid in operations and diagnostics.
	
	● *type=client*
	
	Starts up a classical IP client only. This is the default.
	
	● *type=server*
	
	Starts up a classical IP server. Only one server for each LIS is allowed, and the server needs to be started first.
	
	● *type=(server,client)*
	
	Starts up a classical IP server and client.
	
	Keywords and their meanings for /CLIP are as follows:

<table>
<tr><th>Qualifier</th><th colspan="2">Description</th></tr>
<tr><td></td><td>Keyword</td><td>Meaning</td></tr>
<tr><td></td><td>Create</td><td>Loads the classical IP driver but does not start it.</td></tr>
<tr><td></td><td>Enable</td><td>Causes the node to join the logical IP subnet.</td></tr>
<tr><td></td><td>Disable</td><td>Causes the node to leave the logical IP subnet.</td></tr>
<tr><td>/CONTENDER

/NOCONTENDER</td><td colspan="2">Specifies that the Token Ring device is to participate in the Monitor Contention process when it joins the ring. The /NOCONTENDER qualifier directs the device not to challenge the current ring server.</td></tr>
<tr><td>/DEVICE_SPECIFIC=

FUNCTION="xxxx",

VALUE=n</td><td colspan="2">Allows some device-specific parameters to be adjusted.

The following list shows the commands and their meanings.

FUNCTION="CCOU" – Clears all device and driver counters. If the value is supplied, it is ignored.
FUNCTION="DXMT", VALUE=n – Changes the transmit delay value which is the number of microseconds after completion of a transmit request that an interrupt is generated. The current setting is displayed in the internal counters. This function is applicable to Gigabit NICs.
FUNCTION="DRCV", VALUE=n – Changes the receive delay value which is the number of microseconds after completion of a receive that an interrupt is generated. The current setting is displayed in the internal counters. This function is applicable to Gigabit NICs.
FUNCTION="CXMT", VALUE=n – Changes the transmit coalesce value which is the number of transmit buffer descriptors that are processed before an interrupt is generated. An interrupt may be generated earlier if transmit delay threshold is reached or when an interrupt on behalf of receive or a link state change is generated. The current setting is displayed in the internal counters. This function is applicable to Gigabit NICs.
FUNCTION="CRCV", VALUE=n – Changes the receive coalesce value which is the number of receive buffer descriptors that are filled in before an interrupt is generated. An interrupt may be generated earlier if receive delay threshold is reached or when an interrupt on behalf of transmit or a link state change is generated. The current setting is displayed in the internal counters. This function is applicable to Gigabit NICs.</td></tr>
<tr><td>/DISCOVERY_TIMER= value</td><td colspan="2">Sets the number of seconds to wait for a reply from a remote node when performing the source routing route discovery process.</td></tr>
<tr><td>/DLL=(enable-option, exclusive-option, size-</td><td colspan="2">Provides the MOP downline load service settings for the device.</td></tr>
</table>

Qualifier	Description
`option, knownclientsonly-option)`	In this qualifier, you can specify: ● `enable-option` Indicates that MOP downline load service should be enabled or disabled for the device. ● `exclusive-option` Indicates that no other provider of MOP downline load service is allowed on the specified LAN device at the same time as LANACP. ● `knownclientsonly-option` Indicates that MOP downline load requests should be serviced only for clients defined in the LAN volatile node database. ● `size-option` Specifies the size in bytes of the file data portion of each downline load message.
/EARLY	Enables Early Token Release on the device.

/ELAN [ddag]	Sets LAN emulation. The /ELAN qualifier has two values: enable and disable. With /ELAN=ENABLE with the keyword STARTUP, the LAN emulation is loaded when LANACP starts. With /ELAN=DISABLE, the same parameters as ENABLE can be used.
	The /ELAN qualifier's syntax is as follows:
	`SET DEVICE/ELAN =(parent=parent device,`
	`name="ELAN NAME to join",`
	`size=1516`
	`type=CSMACD`
	`Enable,`
	`Disable,`
	`description = "description string,")`
	The meaning of the syntax for /ELAN are as follows:
	● `parent`
	The ATM adapter device name. An example of the parent device for DGLTA is: HC *n*0, where *n* is the controller number. An example of the parent device for DAPCA is: HW *n*0, where *n* is the controller number.
	● `name`
	Optionally specified if you want to join a specific ELAN. The default is null.

<table>
<tr><th>Qualifier</th><th>Description</th></tr>
<tr><td></td><td>

<code>size</code>

Maximum frame size of the LAN you want to join. Valid sizes are 1516, 4544, or 9234. The default is 1516.
<code>type</code>

Support currently only for CSMACD, which is the default.
<code>description</code>

A method of describing the ELAN for display purposes only.

Keywords and their meanings for /ELAN are as follows:
<table>
<tr><th>Keyword</th><th>Meaning</th></tr>
<tr><td>Create</td><td>Loads the emulation driver SYS $ELDRIVER.EXE, but does not start it.</td></tr>
<tr><td>Enable</td><td>Begins a join on a specified emulated LAN. It also loads the driver, if not already loaded.</td></tr>
<tr><td>Disable</td><td>Causes a client to leave the emulated LAN.</td></tr>
</table>
</td></tr>
<tr><td>/ENABLE</td><td>Enables a LAN device (previously identified as a member of a LAN Failover set) as the active participant in a LAN Failover set.</td></tr>
<tr><td>/FAILOVER_SET=(device-name[,...])

/[NO]FAILOVER_SET=(device-name[,...])</td><td>Specifies the participants of a LAN Failover set.</td></tr>
<tr><td>/FULL_DUPLEX

/NOFULL_DUPLEX</td><td>Enables full-duplex operation of a LAN device. Before full-duplex operation results from the use of this qualifier, additional device or network hardware setup may be required. Some devices may be enabled for full-duplex operation by default. Some devices may not allow the setting to be changed.

The /NOFULL_DUPLEX qualifier disables full-duplex operations.</td></tr>
<tr><td>/JUMBO

/NOJUMBO (default)</td><td>Enables the use of jumbo frames on a LAN device. Only the Gigabit Ethernet NICs support jumbo frames.</td></tr>
<tr><td>/MAP=(MULTICAST_ADDRESS=address,

FUNCTIONAL_ADDRESS=address)</td><td>Defines a functional address mapping entry.</td></tr>
<tr><td>/MAX_BUFFERS= <code>value</code></td><td>Sets the maximum number of receive buffers to be allocated and used by the LAN driver for the LAN device.</td></tr>
</table>

Qualifier	Description
/MEDIA= `value`	• For Token Ring devices: Selects the type of cable that is being used to connect the adapter to the Token Ring Media Access Unit or MAU for devices that do not automatically detect this. • For Ethernet devices: Selects the cable connection.
/MIN_BUFFERS= `value`	Sets the minimum number of receive buffers to be allocated and used by the LAN driver for the LAN device.
/PERMANENT_DATABASE (SET command only)	Updates the device entries in the LAN volatile device database with any data currently set in the permanent database.
/PRIORITY=value	Sets the Failover priority of a LAN device. Priority is given to the LAN Failover participant with the highest priority, when the active participant of a LAN Failover set is chosen.
/PVC=(vci[,...]) [ddag]	Defines the permanent virtual circuit (PVC). This is an optional qualifier. A list of PVCs is defined for use by CLIP clients. This command should be used before enabling the CLIP client. PVC has to be setup manually in the ATM switch. The **vci** is the VCI (Virtual Circuit ID) of the PVC.
/NOPVC=(vci[,...]) [ddag]	Does not set the permanent virtual circuit (PVC).
/RING_PURGER [ddag]	Enables the ring purging process of the FDDI device.
/SOURCE_ROUTING [ddag]	Enables source routing on the Token Ring device.
/SPEED= `value`	Sets the speed of the LAN, if multiple speeds are supported.
/SR_ENTRY=(LAN_ADDRESS= `address`, RI= `routing-information`)	Statically defines a specific source-routed route for a specific node.
/TOKEN_ROTATION [ddag]	Sets the requested token rotation time for the FDDI ring.
/TOKEN_TIMEOUT [ddag]	Sets the restricted token timeout time for the FDDI ring.
/TRANSMIT_TIMEOUT [ddag]	Sets the valid transmission time for the FDDI device.
/UPDATE	Adds LAN devices that are not currently in one of the LAN device databases to that database. The DEFINE DEVICE command applies to the permanent database; the SET DEVICE command applies to the volatile database.
/VOLATILE_DATABASE (DEFINE command only)	Updates the device entries in the LAN permanent device database with any data currently set in the volatile database.

[ddag] Alpha only

Examples

1. `LANCP> SET DEVICE/CONTENDER/MEDIA=UTP/NOEARLY/SOURCE ICA0`

This command enables monitor contention, UTP cable media, and source routing, and disables early token release for Token Ring device ICA0.

2. `LANCP> SET DEVICE/MEDIA=TWIST EWB0`

   This command sets the media type to twisted pair for the second Tulip Ethernet device.

3. `LANCP> SET DEVICE/ALL/MIN_BUFFERS=12`

   This command sets the number of receive buffers for all LAN devices to be no less than 12.

4. `LANCP> DEFINE DEVICE EXA0/MOPDLL=(ENABLE,EXCLUSIVE)`

   This command defines LAN device EXA0 to enable LANACP MOP downline load service in exclusive mode. The settings of the KNOWNCLIENTSONLY and SIZE characteristics are not changed. If the device entry does not currently exist in the LAN permanent device database, these settings will be set to the defaults.

5. `LANCP> DEFINE DEVICE/ALL/MOPDLL=NOEXCLUSIVE`

   This command sets all LAN devices defined in the LAN permanent device database to nonexclusive mode for LANACP MOP downline load service.

6. ```
   LANCP> SET DEVICE EXA0/MOPDLL=(ENABLE,NOEXCLUSIVE)
   LANCP> SET DEVICE FXA0/MOPDLL=(ENABLE,EXCL,KNOWN)
   ```

   These commands enable LANACP MOP downline load service for:

   - LAN device EXA0 in nonexclusive mode
   - LAN device FXA0 in exclusive mode for only known clients

10.6. Managing the LAN Device Databases

The LAN volatile and permanent device databases contain a single entry for each LAN device that exists on the system. Each entry in the LAN volatile device database contains device information and MOP downline load counters information. Each entry in the LAN permanent device database contains device information that is used to populate the volatile database when the LANACP LAN server process is started.

Typically, each database contains the same devices. However, the permanent database may contain entries for devices that have not yet been configured or installed in the system. The LANACP LAN server process maintains the volatile device database. The LANCP utility maintains the permanent device database. You can manipulate either database using the LANCP utility commands depending on your user privileges, as follows:

- Privileged users can add or delete device entries from each database, enable or disable MOP downline load service, and clear MOP downline load counters information for LAN devices.
- Unprivileged users can view the MOP downline load status and counters information.

The following sections describe how to enter and remove devices from the LAN permanent and volatile device databases, and how to enable and disable MOP downline load services.
   ```

10.6.1. Displaying Devices in the LAN Device Databases

To display information in the LAN permanent device database, enter the LIST DEVICE command using the following syntax:

```
LIST DEVICE device-name [/qualifiers]
```

To display information in the LAN volatile device database, enter the SHOW DEVICE command using the following syntax:

```
SHOW DEVICE device-name [/qualifiers]
```

Table 10.10, "LIST DEVICE and SHOW DEVICE Command Qualifiers" provides a brief description of the LIST DEVICE and SHOW DEVICE qualifiers.

Table 10.10. LIST DEVICE and SHOW DEVICE Command Qualifiers

Qualifier	Description
/ALL	Lists, or shows all devices which match device names.
/CHARACTERISTICS	Same as the /PARAMETER qualifier.
/COUNTERS[dag]	Displays device counters.
/DLL	Lists or shows downline load characteristics.
/MAP	Displays the current configuration of the functional address mapping table.
/MOPDLL	Same as DLL.
/OUTPUT= `file-name`	Creates the specified file and directs output to it.
/PARAMETERS	Displays status and related information about the device.
/REVISION [dag]	Displays the current firmware revision of the adapter, if available or applicable.
/SR_ENTRY	Displays the contents of the current source routing cache table.

[dag] SHOW DEVICE only

Note

If you do not specify a qualifier, the utility displays the matching devices without additional information.

10.6.2. Entering Devices into the LAN Device Databases

To enter a device into the LAN permanent device database or to modify an existing entry, enter the DEFINE DEVICE command using the following syntax:

```
DEFINE DEVICE device-name [/qualifiers]
```

To enter a device into the LAN volatile device database or to modify an existing entry, enter the SET DEVICE command using the following syntax:

```
SET DEVICE device-name [/qualifiers]
```

10.6.3. Deleting Devices from the LAN Device Databases

To delete a device from the LAN permanent device database, enter the PURGE DEVICE command using the following syntax:

```
PURGE DEVICE device-name [/ALL]
```

To delete a device from the LAN volatile device database, enter the CLEAR DEVICE command using the following syntax:

```
CLEAR DEVICE device-name [/ALL]
```

For the PURGE DEVICE and CLEAR DEVICE commands, the /ALL qualifier deletes all LAN devices in the LAN permanent device database.

Examples

1. `LANCP> PURGE DEVICE/ALL`

   This command deletes all devices from the LAN permanent device database.

2. `LANCP> CLEAR DEVICE EXA0`

   This command deletes device EXA0 from the LAN volatile device database.

10.7. Managing the LAN Node Databases

The LAN volatile and permanent node databases contain a single entry for each defined LAN node. Each entry in the LAN volatile node database contains node information and MOP downline load counters information. Each entry in the LAN permanent node database contains node information that is used to populate the volatile database when the LANACP LAN server process is started.

Typically, each database contains the same nodes. The LANACP LAN server process maintains the volatile node database. The LANCP utility maintains the permanent node database. You can manipulate either database using the LANCP utility commands depending on your user privileges, as follows:

- Privileged users can add or delete node entries from each database and clear MOP downline load counters information for LAN nodes
- Unprivileged users can view the node information and MOP downline load status and counters information

The following sections describe how to enter nodes into and remove nodes from the LAN permanent and volatile node databases.

10.7.1. Displaying Nodes in the LAN Node Databases

To display information in the LAN permanent node database, enter the LIST NODE command using the following syntax:

```
LIST NODE node-name [/ALL]
```

To display information in the LAN volatile node database, enter the SHOW NODE command using the following syntax:

```
SHOW NODE node-name [/ALL]
```

For the LIST NODE and SHOW NODE commands, the /ALL qualifier displays data for all nodes in the LAN permanent or volatile node database.

10.7.2. Entering Nodes into the LAN Node Databases

To enter a node into the LAN permanent node database or to modify an existing entry, enter the DEFINE NODE command using the following syntax:

```
DEFINE NODE node-name [/qualifiers]
```

To enter a node into the LAN volatile node database or to modify an existing entry, enter the SET NODE command using the following syntax:

```
SET NODE node-name [/qualifiers]
```

Table 10.11, "DEFINE NODE and SET NODE Command Qualifiers" provides a brief description of the DEFINE NODE and SET NODE command qualifiers.

Table 10.11. DEFINE NODE and SET NODE Command Qualifiers

Qualifier	Description
/ADDRESS= `node-address`	Associates a LAN address with the node name.
/ALL	Defines data for all nodes in the LAN permanent or volatile node database.
/BOOT_TYPE= VAX_SATELLITE | ALPHA_SATELLITE |OTHER	Indicates the type of processing required for downline load requests.
/FILE= `file-spec`	Supplies the file name you want to be provided when the downline load request does not include a file name.
/PERMANENT_DATABASE (SET command only)	Updates the node entries in the LAN volatile node database with any data currently set in the permanent database.
/ROOT= `directory-specification`	Supplies the directory specification to be associated with the file name.
/SIZE= `value`	Specifies the size in bytes of the file data portion of each downline load message.
/V3	Forces the server to respond to only MOP Version 3 boot requests from this node.
/VOLATILE_DATABASE (DEFINE command only)	Updates the node entries in the LAN permanent node database with any data currently set in the volatile database.

Examples

1.
```
DEFINE NODE GALAXY/ADDRESS=08-00-2B-11-22-33 -
                   /FILE=NISCS_LOAD.EXE -
                   /ROOT=$64$DIA14: <SYS10.>  -
                   /BOOT_TYPE=VAX_SATELLITE
```

   This command sets up node GALAXY in the LAN permanent node database for booting as a VAX satellite into an OpenVMS Cluster system.

The NISCS_LOAD.EXE file is actually located on 64DIA14: SYS10.SYSCOMMON.SYSLIB. The SYSCOMMON.SYSLIB is supplied by the LANACP LAN server process and is not included in the root definition.

2.
```
DEFINE NODE ZAPNOT/ADDRESS=08-00-2B-11-22-33 -
                    /FILE=APB.EXE -
                    /ROOT=$64$DIA14: <SYS10.>  -
                    /BOOT_TYPE=ALPHA_SATELLITE
```

   This command sets up node ZAPNOT for booting as an Alpha satellite into an OpenVMS Cluster system. The APB.EXE file is actually located on 64DIA14: SYS10.SYSCOMMON.SYSEXE. Note that the SYSCOMMON.SYSEXE is supplied by the LANACP LAN server process and is not included in the root definition.

3.
```
SET NODE CALPAL/ADDRESS=08-00-2B-11-22-33 -
                /FILE=APB_061.EXE
```

   This command sets up node CALPAL for booting an InfoServer image. It defines the file that should be loaded when a load request without a file name is received from node CALPAL.

   Because the file does not include a directory specification, the logical name LAN$DLL defines where to locate the file. You could give a directory specification using the file name or by using the /ROOT qualifier.

   Note that specifying the file name explicitly in the boot command overrides the file name specified in the node database entry.

10.7.3. Deleting Nodes from the LAN Node Databases

To delete a node from the LAN permanent node database, enter the PURGE NODE command using the following syntax:

```
PURGE NODE node-name [/ALL]
```

To delete a node from the LAN volatile node database, enter the CLEAR NODE command using the following syntax:

```
CLEAR NODE node-name [/ALL]
```

For the PURGE NODE and CLEAR NODE commands, the /ALL qualifier deletes all LAN nodes in the LAN permanent or volatile node database.

10.8. Understanding LAN MOP

The collection of utilities and startup command files for LANCP and LANACP provide the necessary functionality for MOP downline load service. These utilities and files load cluster satellites, terminal servers, and systems requiring downline load of special images, such as console update images or system software update images (for InfoServer load).

10.8.1. Coexistence with DECnet MOP

The LAN MOP environment provides functionality that is similar to that provided by DECnet. The result is that a system manager can choose which functionality to use, DECnet MOP or LAN MOP.

For OpenVMS Cluster systems, LAN MOP permits the operation of a cluster without the presence of DECnet.

LAN MOP can coexist with DECnet MOP in the following ways:

- Running on different systems

  For example, DECnet MOP service is enabled on some of the systems on the LAN, and LAN MOP is enabled on other systems.

- Running on different LAN devices on the same system

  For example, DECnet MOP service is enabled on a subset of the available LAN devices on the system, and LAN MOP is enabled on the remainder.

- Running on the same LAN device on the same system but targeting a different set of nodes for service

  For example, both DECnet MOP and LAN MOP are enabled, but LAN MOP has limited the nodes to which it will respond. This allows DECnet MOP to respond to the remainder.

10.8.2. Migrating from DECnet MOP to LAN MOP

To migrate to LAN MOP, follow these steps:

1. Decide which nodes are to provide MOP downline load service. These may be the same nodes that currently have service enabled for DECnet.
2. Populate the LAN permanent device database by typing the following command at the DCL prompt:

   ```
   MCR LANCP DEFINE DEVICE/UPDATE
   ```

3. Populate the LAN permanent node database by entering a node definition for each cluster satellite node and any other nodes that are similarly defined in the DECnet node database. You can enter this data manually or execute the command procedure SYS$EXAMPLES:LAN$POPULATE.COM, following the directions and help provided.
4. Disable service on each of the DECnet circuits where it is currently enabled in the volatile database.
5. Enable service on each LAN device in the LAN permanent device database that you would like to use by typing the following command at the DCL prompt for each device:

   ```
   MCR LANCP DEFINE DEVICE device-name/MOPDLL=ENABLE
   ```

6. If high performance is required, select a data size of 1482 bytes and only reduce this if some load requests now fail. Alternatively, set up one system to load those clients that require a small data size and set up a different system to load the other clients.

To permanently migrate back to DECnet MOP, follow these steps:

1. Disable the MOP service in the volatile database by typing the following:

   ```
   MCR LANCP SET DEVICE device-name/MOPDLL=DISABLE
   ```

2. Disable the MOP service in LANCP's permanent database by typing the following:

   ```
   MCR LANCP DEFINE DEVICE device-name/MOPDLL=DISABLE
   ```

3. Reenable service on each DECnet circuit in the permanent and volatile databases.

Note

Any nodes that you added while booting with LAN MOP will not have been entered in the DECnet node database as targets for downline load, and they will need to be updated when you return to DECnet MOP.

10.8.3. Using CLUSTER_CONFIG_LAN.COM and LAN MOP

A cluster management command procedure has been provided to facilitate the use of LANCP for LAN MOP booting of satellites. Called CLUSTER_CONFIG_LAN.COM, it resides in SYS $MANAGER and is a direct parallel to CLUSTER_CONFIG.COM, which is used by cluster managers to configure and reconfigure an OpenVMS Cluster system. The two procedures perform the same functions, but CLUSTER_CONFIG.COM uses DECnet MOP for downline load, whereas CLUSTER_CONFIG_LAN.COM uses LAN MOP and does not use DECnet for anything. Therefore, when you add a new node, CLUSTER_CONFIG_LAN.COM does not ask for the node's DECnet node name and address. Instead, it queries for an SCS node name and an SCS node ID number.

For your convenience, you can still run CLUSTER_CONFIG.COM. When you execute CLUSTER_CONFIG.COM, it checks whether LANACP for MOP booting is also running. It also checks to see if DECnet is running. If LANACP is running and DECnet is not, then CLUSTER_CONFIG.COM dispatches to CLUSTER_CONFIG_LAN.COM. If CLUSTER_CONFIG.COM discovers that both LANACP and DECnet are running, it asks the user whether LAN MOP booting is being used, and whether it should call CLUSTER_CONFIG_LAN.COM for the user.

10.8.4. Sample Satellite Load

The following example shows how to issue commands to the LANCP utility to enable MOP downline load service and to define node ZAPNOT:

```
set acp/opcom
set device eza0/mopdll=enable
set node ZAPNOT/addr=08-00-2B-33-FB-F2/file=APB.EXE-
            /root=$64$DIA24:<SYS11.>/boot=Alpha
```

The following example shows the OPCOM messages displayed when you start up the LANACP LAN server process:

```
%%%%%%%%%%  OPCOM  10-JAN-2017 06:47:35.18  %%%%%%%%%%
Message from user SYSTEM on GALAXY
LANACP MOP Downline Load Service
Found LAN device EZA0, hardware address 08-00-2B-30-8D-1C

%%%%%%%%%%  OPCOM  10-JAN-2017 06:47:35.25  %%%%%%%%%%
Message from user SYSTEM on GALAXY
LANACP MOP Downline Load Service
Found LAN device EZB0, hardware address 08-00-2B-30-8D-1D

%%%%%%%%%%  OPCOM  10-JAN-2017 06:47:54.80  %%%%%%%%%%
Message from user SYSTEM on GALAXY
LANACP MOP V3 Downline Load Service
Volunteered to load request on EZA0 from ZAPNOT
```

```
Requested file:  $64$DIA24:<SYS11.>[SYSCOMMON.SYSEXE]APB.EXE

%%%%%%%%%%  OPCOM  10-JAN-2017 06:48:02.38  %%%%%%%%%%
Message from user SYSTEM on GALAXY
LANACP MOP V3 Downline Load Service
Load succeeded for ZAPNOT on EZA0
System image, $64$DIA24:<SYS11.>[SYSCOMMON.SYSEXE]APB.EXE (Alpha image)
```

The following display shows the contents of the LAN$ACP.LOG file:

```
10-JAN-2017 06:47:35.02  Found LAN device EZA0, hardware address
08-00-2B-30-8D-1C
10-JAN-2017 06:47:35.18  Found LAN device EZB0, hardware address
08-00-2B-30-8D-1D
10-JAN-2017 06:47:35.25  LANACP initialization complete
10-JAN-2017 06:47:45.39  Enabled LAN device EZA0 for MOP downline load
 service in
exclusive mode
10-JAN-2017 06:47:54.70  Volunteered to load request on EZA0 from ZAPNOT
    Requested file:  $64$DIA24:<SYS11.>[SYSCOMMON.SYSEXE]APB.EXE
10-JAN-2017 06:48:02.23  Load succeeded for ZAPNOT on EZA0
    MOP V3 format, System image,
 $64$DIA24:<SYS11.>[SYSCOMMON.SYSEXE]APB.EXE
    Packets:  2063 sent, 2063 received
    Bytes:    519416 sent, 4126 received, 507038 loaded
    Elapsed time:  00:00:07.42, 68276 bytes/second
```

10.8.5. Cross-Architecture Booting

The LAN enhancements permit cross-architecture booting in a OPENVMS Cluster system. VAX boot nodes can provide boot service to Alpha satellites, and Alpha boot nodes can provide boot service to VAX satellites. Note that each architecture must include a system disk that is used for installations and upgrades.

10.9. Managing the LAN MOP Downline Load Service

The LANACP LAN server process maintains the LAN volatile node and device databases. The LANCP utility provides commands that:

- Display MOP downline load status and counters information
- Clear counters information
- Enable or disable OPCOM messages and packet tracing

Counters and status information is maintained for each node and device. Counters information includes transmitted and received byte and packet counts, transmit errors, logical errors such as protocol violations and timeouts, and number of load requests. Status includes the time of the last load and the status of the last load.

10.9.1. Enabling MOP Downline Load Service

To enable MOP downline load service, enter the SET DEVICE command using the following syntax:

```
SET DEVICE device-name/DLL=ENABLE
```

In this command, use the `device-name` parameter to supply the LAN controller device name.

See *Section 10.6.2, "Entering Devices into the LAN Device Databases"* for a complete description of this command.

10.9.2. Disabling MOP Downline Load Service

To disable MOP downline load service, enter the SET DEVICE command using the following syntax:

```
SET DEVICE device-name/DLL=DISABLE
```

In this command, use the `device-name` parameter to supply the LAN controller device name.

See *Section 10.6.2, "Entering Devices into the LAN Device Databases"* for a complete description of this command.

10.9.3. Displaying the Status and Counters Data

To display MOP downline load status, enter the SHOW DLL command using the following syntax:

```
SHOW DLL
```

The following display shows counters information for a particular node:

```
LAN MOP DLL Status:
   EXA enabled in exclusive mode for known nodes only, data size 1482 bytes
   FXA disabled

        #Loads   Packets        Bytes     Last load time       Last loaded
        ------   -------        -----  --------------------
 ------------------
   EXA      5      1675      4400620  10-JAN-2017 10:27.51  GALAXY
   FXA      0         0            0
```

On this node are two LAN devices, EXA (DEMNA) and FXA (DEMFA). MOP downline load service is enabled on EXA in exclusive mode.

Requests are answered only for nodes that are defined in the LANACP node database. The image data size in the load messages is 1482 bytes. There have been five downline loads, the last one occurring on node GALAXY at 10:27. Finally, no downline loads are recorded for FXA, which is currently disabled for downline load service.

To display recent downline load activity that has been logged in the LAN$ACP.LOG file, enter the SHOW LOG command using the following syntax:

```
SHOW LOG
```

10.9.4. Displaying the Status and Counters Data for Individual Nodes

To display MOP downline load information for nodes in the LAN permanent node database, enter the LIST NODE command using the following syntax:

```
LIST NODE node-name [/qualifiers]
```

To display MOP downline load status and counters information for nodes in the LAN volatile node database, enter the SHOW NODE command using the following syntax:

```
SHOW NODE node-name [/qualifiers]
```

Table 10.12, "LIST NODE and SHOW NODE Command Qualifiers" provides a brief description of the LIST NODE and SHOW NODE command qualifiers.

Table 10.12. LIST NODE and SHOW NODE Command Qualifiers

Qualifier	Description
/ALL	Displays information for all nodes in the database.
/OUTPUT= `file-name`	Indicates that the output should be directed to the specified file. If the file name extension is .com, then the output is in the form of a list of DEFINE NODE or SET NODE commands. The resulting command file can be used to create the LAN node databases.
/TOTAL (SHOW NODE command only)	Displays counter totals only.

Example

The following example shows output from a command issued on a local node on which there are three nodes defined (GALAXY, ZAPNOT, and CALPAL). CALPAL has issued two load requests:

- The first request is the multicast request from CALPAL that the local node volunteered to accept.
- The second request is the load request sent directly to the local node by CALPAL for the actual load data. The elapsed time from the second load request to completion of the load was 6.65 seconds.

```
Node Listing:

GALAXY (08-00-2B-2C-51-28):
  MOP DLL:  Load file:   APB.EXE
            Load root:   $64$DIA24:<SYS11.>
            Boot type:   Alpha satellite

ZAPNOT (08-00-2B-18-7E-33):
  MOP DLL:  Load file:   NISCS_LOAD.EXE
            Load root:   LAVC$SYSDEVICE:<SYS10.>
            Boot type:   VAX satellite

CALPAL (08-00-2B-08-9F-4C):
  MOP DLL:  Load file:   READ_ADDR.SYS
            Last file:   LAN$DLL:APB_X5WN.SYS
            Boot type:   Other
            2 loads requested, 1 volunteered
            1 succeeded, 0 failed
            Last request was for a system image, in MOP V4 format
            Last load initiated 10-jan-2017 09:11:17 on EXA0 for
 00:00:06.65
            527665 bytes, 4161 packets, 0 transmit failures

Unnamed (00-00-00-00-00-00):

Totals:
  Requests received    2
```

```
  Requests volunteered 1
  Successful loads     1
  Failed loads         0
  Packets sent         2080
  Packets received     2081
  Bytes sent           523481
  Bytes received       4184
  Last load            CALPAL at 10-jan-2017 09:11:17.29
```

10.9.5. Clearing the Counters Data

To clear MOP downline load counters for all nodes and devices, enter the CLEAR DLL command using the following syntax:

```
CLEAR DLL
```

10.9.6. OPCOM Messages

By default, OPCOM messages are enabled. Messages are generated by the LANACP LAN server process when device status changes, load requests are received, and loads complete. These messages are displayed on the operator's console and included in the log file written by LANACP, SYS $MANAGER:LAN$ACP.LOG.

To enable OPCOM messages, enter the SET ACP/OPCOM command using the following syntax:

```
SET ACP/OPCOM
```

10.9.7. Load Trace Facility

If the error data produced by the LANACP LAN server process for a load request is not sufficient to help you determine why the load is failing, you can direct the server process to record trace data. The data consists of transmit and receive packet information for every transmit and receive done by the server, and written to a log file for each load attempt. The name of the log file is SYS $MANAGER:LAN$ `nodename`.LOG. You can record either all packet data or only the first 32 bytes of each packet.

The following list describes the typical load sequence:

1. Receive a Program Request message on the Load Assistance Multicast Address from the requesting node, code 8.
2. Transmit an Assistance Volunteer message to the requesting node, code 3.
3. Receive a Program Request message on your node address from the requesting node, code 8.
4. Transmit a Memory Load message to the requesting node with sequence number zero, code 2.
5. Receive a Request Memory Load message requesting the next sequence number (modulo 256), code 10 (decimal).
6. Repeat steps 4 and 5 until there is no more data to send.
7. Transmit a Memory or Parameter Load with Transfer Address message, code 0 or 20 (decimal).
8. Receive a final Request Memory Load message requesting the next sequence number (modulo 256) indicating that the last message has been received, code 10 (decimal).

For cluster satellite loads, the last Memory Load message contains cluster parameters. This message and the final Load with Transfer Address messages are displayed in full even if only partial trace echo has been enabled.

To enable partial tracing of packet data, enter the SET ACP/ECHO command using the following syntax:

```
SET ACP/ECHO
```

To enable full tracing of packet data, add the /FULL qualifier:

```
SET ACP/ECHO/FULL
```

10.9.8. MOP Console Carrier

Console carrier provides a mechanism to connect to a LAN device, such as a terminal server, that implements a management interface using the MOP console carrier protocol. The LANCP utility provides this function in the form of a CONNECT NODE command.

The command syntax is:

```
CONNECT NODE node-specification [/qualifiers]
```

Table 10.13, "CONNECT NODE Command Qualifiers" provides a brief description of the CONNECT NODE command qualifiers.

Table 10.13. CONNECT NODE Command Qualifiers

Qualifier	Description
/DEVICE= `device-name`	Specifies the LAN controller device name to be used for the connection.
/DISCONNECT= `disconnect-character`	Specifies a character that you can use to terminate the connection to the remote node.
/PASSWORD=16hexdigits	Supplies the password to be used when the connection is initiated.
/V3 or /V4	Indicates that MOP Version 3 or Version 4 formatted messages, respectively, are to be used to make the connection.

Examples

1. `CONNECT NODE GALAXY/DEVICE=EWA0`

   This command attempts a console carrier connection to node GALAXY using the Ethernet device EWA0.

2. `CONNECT NODE 08-00-2B-11-22-33/DEVICE=EWA0/PASSWORD=0123456789ABCDEF`

   This command attempts a console carrier connection to the given node address using the Ethernet device EWA0, with a password.

10.9.9. MOP Trigger Boot

Some systems recognize and respond to MOP remote boot requests. These systems typically require a password or other mechanism to prevent unwanted boot requests from triggering a reboot of the system. The LANCP utility provides this function in the form of the TRIGGER NODE command.

To request a reboot of a LAN system, enter the TRIGGER NODE command using the following syntax:

```
TRIGGER NODE node-specification [/qualifiers]
```

Table 10.14, "TRIGGER NODE Command Qualifiers" provides a brief description of the TRIGGER NODE command qualifiers.

Table 10.14. TRIGGER NODE Command Qualifiers

Qualifier	Description
/DEVICE= `device-name`	Specifies the LAN controller device name to be used for sending the boot messages.
/PASSWORD=16hexdigits	Supplies the password to be used when the connection is initiated.

Rather than specify the format to send MOP Version 3 or 4, the LANCP utility sends one message in each format to the target node.

The following examples show how to use the TRIGGER NODE command:

Examples

1. `TRIGGER NODE GALAXY/DEVICE=EWA0`

   This command sends MOP trigger boot messages to node GALAXY using Ethernet device EWA0.

2. `TRIGGER NODE 08-00-2B-11-22-33/DEVICE=EWA0/PASSWORD=0123456789ABCDEF`

   This command sends MOP trigger boot messages to the given node address using the Ethernet device EWA0, with indicated password.

10.10. Understanding LAN Failover

LAN Failover is a mechanism for protecting your system from a network interface card (NIC) failure. LAN Failover does this by integrating individual network adapters on the same local network into a single virtual interface called a LAN Failover set.

The system manager defines and creates a LAN Failover set using LANCP utility program. A failover set consists of one adapter that is used for LAN traffic and one or more adapters that remain idle. If the active adapter fails, one of the idle set members automatically takes over, allowing for continuous operation.

Software: LLc and Lldriver

LAN Failover integrates multiple network devices on the same LAN segment into a failover set and presents it as a single virtual device, LLc, to the LAN applications (where c is a user-specified alphabetic character uniquely identifying the LAN virtual device).

The virtual LAN driver, Lldriver, supports LAN Failover. Lldriver provides the standard IEEE 803 interfaces to the higher software layers, maintains the logical groupings of all the adapters in a failover set, and transfers I/O to the active physical port driver.

One network device in the set is active while the others remain idle. If the active device fails, Lldriver selects one of the idle devices from the failover set to become active. The physical address and multicast addresses of the virtual device remain the same, allowing for continuous operation of LAN applications.

Hardware Requirements

LAN Failover requires that redundant network interface cards be on the same local area network. The network interface cards supported by LAN Failover are the DEGPA, the DEGXA, and the DE600 series.

Detecting Network Connectivity Failures

LAN Failover provides a mechanism to protect against certain kinds of network connectivity failures. The virtual and physical LAN drivers detect network interface card (NIC) failures and failures that exist between the NIC and the switch.

Restrictions for Using LAN Failover

Restrictions for using LAN Failover are the following:

- LAN Failover is not supported on cluster satellites.
- Network interface cards connected point-to-point are not supported in a LAN Failover set.
- Physical devices cannot be in use when they are added to a LAN Failover set.

Managing LAN Failover

The LAN volatile and permanent device databases contain a single entry for each LAN device that exists on the system. The virtual device, LLc, is added to these databases through LANCP when the system manager creates a LAN Failover set.

To enter the LLc device into the LAN permanent device database or to modify an existing entry, enter the LANCP command DEFINE DEVICE LLc using the following syntax:

```
DEFINE DEVICE LLc[/qualifiers]
```

This command allows the settings to take effect on subsequent boots.

Before a physical adapter can be added to a failover set, all users on that adapter must be stopped.

10.10.1. Creating a LAN Failover Set

The virtual device, LLc, is created when you create a LAN Failover set. To create a LAN Failover set, enter the LANCP command SET DEVICE LLc using the following syntax:

```
SET DEVICE LLc/FAILOVER_SET=(device-name[,...])
```

In this command, supply the LAN physical device name for the device-name; for example:

```
LANCP> SET DEVICE LLA/FAILOVER_SET=(EWA,EIA,EWB)
```

This command fails if any specified LAN devices have active users.

10.10.2. Removing a LAN Failover Set

To remove a LAN Failover set, enter the LANCP command SET DEVICE LLc using the following syntax:

```
SET DEVICE LLc/NOFAILOVER_SET=(device_name[,...])
```

In this command, supply the LAN physical device name for the device-name; for example:

```
LANCP> SET DEVICE LLA/NOFAILOVER_SET=(EWB)
```

This command fails if one of the specified devices is the active physical device.

10.10.3. Setting the Priority of a LAN Failover Participant

To give preference to a physical LAN device when you select the active participant of a LAN Failover set, enter the LANCP command SET DEVICE/PRIORITY command using the following syntax:

```
SET DEVICE device-name/PRIORITY=value
```

In this command, supply the LAN physical device name for the device-name and an integer for the value parameter; for example:

```
LANCP> SET DEVICE EIA/PRIORITY=20
```

When selecting the active participant, the system gives preference to the device with the highest priority.

10.10.4. Enabling LAN Failover

Enabling a LAN Failover device establishes the physical address of the virtual device and selects the active participant of the LAN Failover set.

To enable LAN Failover, enter the LANCP command SET DEVICE LLc using the following syntax:

```
SET DEVICE LLc/ENABLE
```

10.10.5. Disabling LAN Failover

Disabling a LAN Failover device disassociates the virtual device from a physical device. This allows all physical devices to be removed from the LAN Failover set.

To disable LAN Failover, enter the LANCP command SET DEVICE using the following syntax:

```
SET DEVICE LLc/DISABLE
```

This command fails if the virtual device has active users.

10.10.6. Displaying LAN Failover Characteristics

To display LAN Failover status, enter the LANCP command SHOW DEVICE LLc using the following syntax:

```
SHOW DEVICE LLc/CHARACTERISTICS
```

The display shows the characteristics that are specific to LAN Failover for a particular node, as shown in the following example:

```
Device Characteristics LLAO:
Value Characteristic
```

```
.
.
.
"EIA" Failover device (active)
"EWA" Failover device
Enabled/Active Logical LAN state
```

10.10.7. Validating a LAN Failover Set

The network devices associated with a LAN Failover set must provide physically redundant paths on the same local network for LAN Failover to function correctly. Since networks are highly stable, the active member of a LAN Failover set might not change often. However, when the active member does change, it is crucial that the idle NICs have been set up properly.

The system manager can validate each member of the failover set by simulating LAN failures using the LANCP qualifier /SWITCH. The /SWITCH qualifier simulates a network failure on the active device and selects another device from the failover set to be the active device.

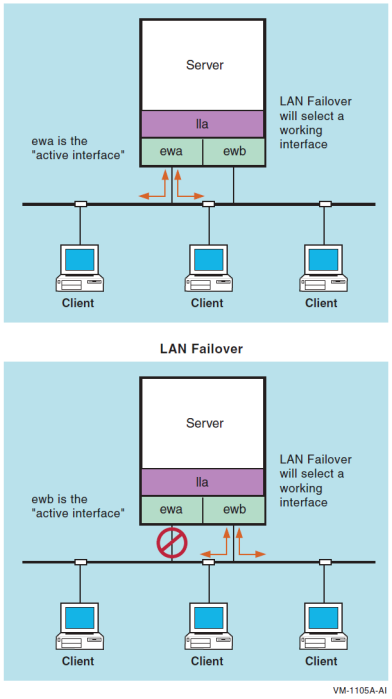
To simulate a LAN failure, enter the LANCP command SET DEVICE LLc using the following syntax:

```
SET DEVICE LLc/SWITCH
```

10.10.8. Illustration of LAN Failover

The following figure illustrates LAN Failover.

Figure 10.1. LAN Failover



Chapter 11. Managing InfoServer Systems

This chapter describes InfoServer functions and InfoServer Client for OpenVMS software, which enables OpenVMS systems to access InfoServer device services. The chapter also describes the tasks you must perform to start the client software on your system and to make InfoServer devices available as public devices.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Establishing a server management session	*Section 11.3, "Establishing a Server Management Session"*
Starting InfoServer Client for OpenVMS software automatically	*Section 11.5.3, "Starting InfoServer Client for OpenVMS Software Automatically"*
Making InfoServer devices available automatically	*Section 11.6.3, "Making InfoServer Devices Available Automatically"*

This chapter explains the following concepts:

Concept	Section
InfoServer functions	*Section 11.1, "Understanding InfoServer Functions"*
LASTport protocols	*Section 11.2, "Understanding LASTport Protocols"*
InfoServer Client for OpenVMS functions	*Section 11.4, "Understanding InfoServer Client for OpenVMS Functions"*
LASTCP utility functions	*Section 11.5, "Understanding*

Concept	Section
	LASTCP Utility Functions"
LADCP utility functions	*Section 11.6, "Understanding LADCP Utility Functions"*

11.1. Understanding InfoServer Functions

The InfoServer system is a high-performance **virtual device server**. It can make available, or **serve**, compact discs, read/write disks, magneto-optical (MO) devices, and tapes to client systems on the local area network (LAN). Systems running InfoServerClient software can connect to the virtual devices and use them as though they are locally attached devices.

Unlike a file server, the InfoServer system does not impose a file system on the virtual devices that it serves. For example, the InfoServer system can serve a disk with any type of on-disk file structure. The client system interprets the on-disk structure and uses its own native file system to access data. Multiple on-disk structures can be served by and accessed on a single InfoServer system at the same time.

The InfoServer system can perform the following functions:

- Serve compact discs

  The InfoServer system serves compact discs automatically, using a disc's volume label as the service name when the server is booted or when a disc is inserted into an InfoServer drive. You do not have to perform any management action. Client systems simply bind to and mount the disc under its volume label.

  The InfoServer system can automatically serve to OpenVMS clients compact discs that are in ODS-2 format. High Sierra and ISO-9660 compact discs and other media types can be served manually through the InfoServer management interface.

- Serve Small Computer System Interface (SCSI) tapes

  Using service names, the InfoServer system can serve SCSI tape devices to the network. Client systems can connect to these tape devices and use them as though they were locally attached devices.

- Serve read/write disk partitions

  A **partition** is a logical subset of an InfoServer read/write disk. A single disk can be subdivided into several partitions, each of which can be served to the network independently. To remote client systems, these partitions appear to be whole disks. For example, a client system using InfoServerClient for OpenVMS software can access the partitions and use them as though they are local hard disks.

- Act as an initial load system for OpenVMS systems

  The InfoServer system can downline load the primary bootstrap program to OpenVMS systems by responding to Maintenance Operation Protocol (MOP) requests. The server can locate MOP downline load files on the OpenVMS software distribution compact disc and copy them into temporary MOP partitions on an InfoServer-formatted read/write disk.

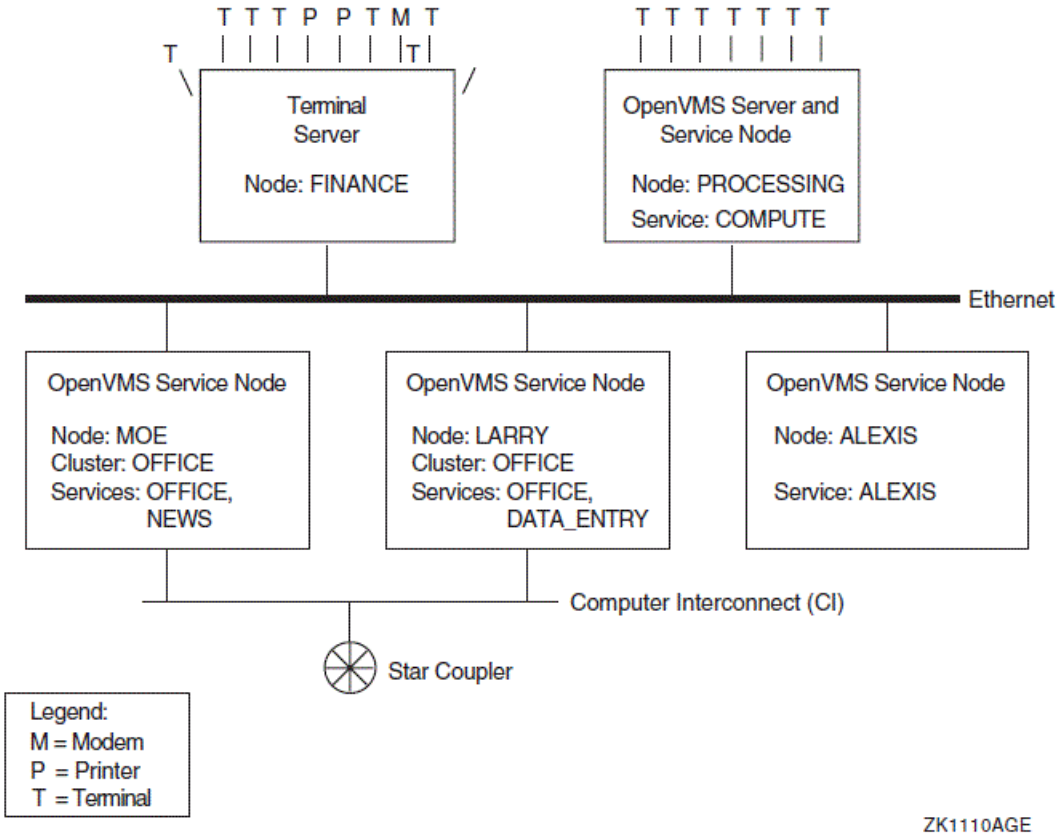
The initial system load (ISL) bootstrap program connects back to the software distribution compact disc and boots Standalone Backup. The Backup utility is then used to copy the OpenVMS operating system save sets from the compact disc to a read/write disk attached to the system. All subsequent OpenVMS boots are done from the local read/write disk.

- Downline load other products

  You can use the InfoServer system to load any Ethernet product by file name;that is, the server does not require a Network Control Program (NCP) database entry to locate the requested file. For example, X terminal clients use the InfoServer system to downline load their system software. You can create a special MOP partition and copy the desired file to that partition. The server additionally supports downline loading of services by Ethernet address. Each InfoServer system can handle up to 100 simultaneous downline loads more efficiently than host-based downline loaders, which must start processes to assist in the load.

Figure 11.1, "InfoServer System Serving Clients" shows the relationship of the InfoServer system to several possible client systems. In this figure, two compact discs and two hard disks connected to the server appear to the client systems as local devices. The VAX system and the RISC workstation might be using one or two of the compact discs for software distribution and online documentation, while the PC might be referencing a disk partition on the InfoServer system. The X terminal boots from the InfoServer system and uses InfoServer disks for page, font, and customization files.

Figure 11.1. InfoServer System Serving Clients



You can connect the InfoServer system to your Ethernet LAN and turn on the system. After the server is initialized, or **bootstrapped**, the server software automatically serves to client systems the device media connected to it. If you insert a compact disc into a server drive, the server detects this new device and automatically serves it to client systems by using the volume label as the service name.

The server bootstraps from its internal read/write device, on which the InfoServer software is preinstalled. InfoServer software updates are distributed on compact discs. As these new releases become available, you can install the software onto the internal device for subsequent booting. To update InfoServer software from the compact disc, follow these steps:

1. Insert the disc in a compact disc drive attached to the InfoServer system.

2. Move the InfoServer software to the internal read/write device. At the InfoServer prompt, enter a command in the following format, where *n* is the drive number:

   On the InfoServer 100 or InfoServer 150 system:

   ```
   InfoServer> UPDATE SYSTEM DKn:
   ```

   On the InfoServer 1000 system:

   ```
   InfoServer> UPDATE SYSTEM DKn: FLASH
   ```

   The next time you boot the InfoServer system, it runs the updated software.

You can use the Software Products Library (formerly known as ConDIST) to update InfoServer software. After you log in to the InfoServer system, perform the following steps:

1. Insert the disk containing the [INFOSERVxxx] directory tree in a compact disk drive attached to the InfoServer system.

2. At the InfoServer> prompt, enter a command in the following format, where `n` is the drive number:

   - On the InfoServer 100 or InfoServer 150 system, enter a command in the following format:

     ```
     UPDATE SYSTEM DKn:
     ```

   - On the InfoServer 1000 system, enter a command in the following format:

     ```
     UPDATE SYSTEM DKn: FLASH
     ```

   These commands move the InfoServer software to the internal read/write device. The next time you boot the InfoServer system, it runs the updated software. Note that you can also boot the server from the Software Products Library disk.

You might want to customize server features. You can control InfoServer functions by logging in to the server and entering server commands, described in detail in the *InfoServer System Operations Guide*.

11.1.1. Automatic Service Policies for Multiple Servers

The InfoServer system automatically serves its locally connected devices to clients when the server is first powered on or when a removable device (for example, a compact disc) is inserted into a drive. The server reads the volume label of each device and uses the label as the name of the service offered to clients.

Note

You can disable the automatic service feature by using the InfoServer command SET SERVER AUTOMOUNT.

If multiple servers offer the same services, the client uses a rating scheme to select the appropriate service. Refer to the CREATE SERVICE command description in the *InfoServer System Operations Guide* for more information.

When you remove a compact disc from a server drive, the InfoServer system ends all client connections to the associated service. The InfoServer system also stops offering the associated service to client systems.

11.1.2. High-Availability Feature to Reduce Service Interruptions

The InfoServer system provides a high-availability feature that is especially beneficial for OpenVMS clients. If the server ends a service connection for some reason (for example, the server reboots, or you remove a compact disc), the OpenVMS client enters mount verification for that volume. If the same service is offered by another InfoServer system on the LAN, the client automatically connects to that service.

For example, suppose you have two identical copies of the OpenVMS Online Documentation compact disc in drives on two different servers. If one server or drive fails, a new connection is established to the duplicate disc on the other server. File operations continue as normal, and users experience almost no service disruption.

11.1.3. Support for X Terminal Clients

X terminal clients use the InfoServer system to download their system software, provide font services, save configuration information, and page memory to and from InfoServer disks. For example, system files for VSI's VXT 2016 windowing terminals can be installed from compact disc on the InfoServer system. Once installed, these files are downline loaded on demand to each terminal when it is powered on.

The terminals can dynamically allocate partitions on an InfoServer disk as needed. For example, when a user requests that terminal customizations be saved, the InfoServer system automatically creates a disk partition to hold the information and creates a network service name for that partition. Once customization information is saved, the user can recall the information at any time.

VXT 2016 terminals that are InfoServer clients can also be virtual memory machines. Such terminals can page sections of main memory to and from InfoServer disks as required. Because a VXT client has no local disk, it uses InfoServer disks as page disks. When main memory is paged out to disk, the VXT client requests the InfoServer system to create a partition. This partition is then automatically extended as needed. Partitions and their network service names are created dynamically, without requiring user action.

By default, the InfoServer disk DK1, which is the internal disk that ships with each InfoServer 150 system, is enabled to allow VXT 2016 clients to allocate partitions remotely. Other disks can also be enabled through the use of InfoServer commands.

11.2. Understanding LASTport Protocols

The InfoServer system uses the LASTport transport protocol and the LASTport/Disk and LASTport/Tape system application protocols to provide access to the virtual devices it serves to the LAN. These protocols provide high-performance access to disk and tape devices. The InfoServer system implements

the server portion of the protocols, while the client systems that access InfoServer storage devices implement the client portion.

On OpenVMS systems running the LASTport transport, all Ethernet devices must be terminated either by attaching the devices to an active network or by using an appropriate terminator. Failure to terminate the devices causes a system crash.

11.2.1. LASTport Transport Protocol

The **LASTport protocol** is a specialized LAN transport protocol that allows many clients to access InfoServer systems and perform reliable transactions. For the InfoServer system, a transaction is a device read or write operation. The LASTport protocol allows many client systems concurrently to read information from, and write information to, an InfoServer storage device.

Unlike timer-based protocols, the LASTport protocol is a transaction-oriented protocol. Normally, information does not pass between a client and an InfoServer system unless the client initiates a transaction. The client system then runs a timer on the transaction, normally waiting from two to five seconds before assuming that the transaction is lost and retrying the operation.

The LASTport protocol does not provide any routing functions; it runs only in a LAN. The LASTport protocol type is 80–41. If the extended LAN uses any filtering devices, the devices must allow this protocol type to pass unfiltered so that clients can access InfoServer systems across the filtering device.

The InfoServer system uses a multicast address feature of the LASTport protocol to establish connections to devices. The format of the multicast address is 09–00–2B–04--`nn-nn`, where `nn` depends on the work group enabled (refer to the *InfoServer System Operations Guide*).

11.2.2. LASTport/Disk Protocol

The LASTport/Disk protocol is a specialized device protocol that uses the LASTport transport. That is, LASTport/Disk messages are delivered in LASTport messages. The LASTport/Disk protocol provides the mechanism for reading and writing logical blocks independent from any underlying file system. The clients that implement the LASTport/Disk protocol interpret the file system locally. By using the LASTport/Disk protocol for access to compact discs and read/write disks, the InfoServer system can support multiple client operating systems and on-disk structures concurrently.

The LASTport/Disk protocol also provides the naming facility to access compact discs and read/write disks. The InfoServer system assigns each virtual device a service name and allows clients to query the LAN for these names. When the requested service is found, the client connects to it, and device access can begin. When duplicate virtual devices are available under identical service names, the protocol provides a facility for load balancing among the available devices.

11.2.3. LASTport/Tape Protocol

Like the LASTport/Disk protocol, the LASTport/Tape protocol uses the LASTport transport. That is, LASTport/Tape messages are delivered in LASTport messages. The LASTport/Tape protocol provides the mechanism for reading and writing tape records. Tape devices attached to the InfoServer system appear to tape clients as locally attached devices.

The LASTport/Tape protocol also provides the naming facility to access tapes. The InfoServer system assigns each tape device a service name and allows clients to query the LAN for these names. When the requested service is found, the client connects to it, and tape access can begin.

11.3. Establishing a Server Management Session

You can establish a server management session from a local or remote console terminal:

- *For a local session,* you connect a terminal capable of interpreting VT100 ANSI escape sequences to the serial port on the rear of the InfoServer system unit (MMJ1 on the InfoServer 150 unit). The terminal must be set to 9600 baud, 8 bits, no parity.
- *For a remote session,* you make a connection to the InfoServer system through a local area terminal (LAT) server.

  Like many network servers, the InfoServer system advertises a LAT service for its management interface and accepts connections from remote terminals attached to terminal servers. Therefore, any terminal attached to a terminal server on the extended LAN can act as a console terminal for the InfoServer system (if the user knows the InfoServer management password).

Determining the Server's Default Service Name

To make a remote connection to the InfoServer system for the first time, you must determine the server's default name. To do this, add the four-character prefix LAD_ to the hexadecimal Ethernet data link address on the InfoServer system's cabinet. You can change this default name by using the InfoServer command SET SERVER NAME.

The server's name is the LAT service name to which you connect. For example, if the default server name is LAD_08002B15009F, you would enter the following command at the terminal server's prompt to manage the InfoServer system:

```
Local> CONNECT LAD_08002B15009F
```

Refer to your terminal server user's guide for information about the establishment of LAT service connections.

Entering an InfoServer Password

After you connect to the InfoServer system, you must enter an InfoServer password to establish the management session. The default server password is ESS. You can change the password with the InfoServer command SET SERVER PASSWORD.

Example

The following example shows the establishment of a sample session using a DECserver 500 terminal server:

```
Local> CONNECT LAD_08002B133C1C
Password: ESS (not echoed)
Local -010- Session 1 to LAD_08002B133C1C established
DEC InfoServer V3.1
InfoServer> SHOW SERVER
```

In this example, the terminal server's prompt is Local>, and a LAT session is established to the InfoServer system whose service name is LAD_08002B133C1C. The InfoServer system prompts for

a server password. When you enter the correct password, the server prompts for InfoServer commands with the InfoServer> prompt.

Ending a Session

At the end of the management session, you can enter the EXIT command at the InfoServer> prompt. This command returns you to the terminal server's Local>prompt if the management session is over a LAT connection.

11.3.1. Server Management Commands

Table 11.1, "Summary of InfoServer Commands" summarizes InfoServer commands and their functions.

Table 11.1. Summary of InfoServer Commands

Command	Function
BACKUP	Saves InfoServer-formatted disks.
BIND	Establishes a connection to the specified ODS-2 service and creates the virtual device VDK1 for that service.
CLEAR	Erases the console terminal screen.
COPY	Copies data from one disk or partition to another.
CRASH	Causes the server software to take a recognizable bugcheck, creating a dump if crash dump processing is enabled.
CREATE	Creates a new partition or service.
DELETE	Deletes a partition or service that was previously created.
DISCONNECT	Terminates a LAST port or LAT terminal server session.
ERASE	Erases the specified disk or partition; erases FUNCTIONS or SERVICES data from nonvolatile random-access memory (NVRAM).
EXIT	Terminates the management session.
HELP	Displays help text for the InfoServer commands.
INITIALIZE	Formats a read/write disk into an InfoServer disk.
LOOP	Automatically repeats any valid InfoServer command.
MONITOR	Automatically repeats valid InfoServer commands every 3 seconds, clearing the screen and placing the cursor at the home position.
PURGE	Purges old versions of VXT software.
REBOOT	Shuts down and reboots the server.
RECORD	Records data from an InfoServer disk or partition to a compact disc.
RESTORE	Resets the server to a previously saved system configuration.
RETRIEVE	Restores InfoServer-formatted disks saved by the BACKUP command.
REWIND	Rewinds an InfoServer tape.
SAVE	Saves configuration and service data for recovery after a server reboot.
SET	Sets partition, service, or server parameters.
SHOW	Displays the server's parameters and counters.
UNBIND	Deletes the VDK1 virtual device and terminates the connection to the remote service.

Command	Function
UNLOAD	Rewinds and unloads an InfoServer tape.
UPDATE	Installs one or more new software products or functions.
VERIFY	Validates the on-disk structure of a device formatted with the INITIALIZE command.
ZERO	Sets internal server counters to 0.

The InfoServer system provides a Help facility that contains information about each server command, including parameters, qualifiers, and examples of its use. For detailed information about InfoServer commands, refer to the *InfoServer System Operations Guide*.

11.4. Understanding InfoServer Client for OpenVMS Functions

InfoServer Client for OpenVMS software enables clients running the OpenVMS operating system to access virtual device services offered by InfoServer systems on a LAN. Software components include the following ones:

- LASTport driver

  The LASTport driver provides reliable data transfer services for its clients. It interacts with the Data Link driver and the LASTport/Disk driver as an efficient transport for a virtual device service. The LASTport driver can support other applications, such as a primitive data queueing service.

- LASTport/Disk client driver

  The LASTport/Disk client driver presents a standard block device interface to the system. The OpenVMS file system interacts with the LASTport/Disk client as if the LASTport/Disk client were a local disk driver. The LASTport/Disk client driver supports both raw and buffered interfaces.

- LASTport/Tape client driver

  The LASTport/Tape client driver enables OpenVMS clients to access and use as local devices SCSI tapes attached to InfoServer systems.

- LASTCP and LADCP utilities

  These utilities allow you to start InfoServer Client software on your system, monitor transport status, and configure and maintain InfoServer device services. *Section 11.5, "Understanding LASTCP Utility Functions"* and *Section 11.6, "Understanding LADCP Utility Functions"*introduce the utilities. For complete information about the utilities, refer to the *InfoServer Client for OpenVMS LASTCP and LADCP Utilities* manual.

11.5. Understanding LASTCP Utility Functions

InfoServer Client for OpenVMS software uses the LASTport protocol to communicate with InfoServer systems on an extended LAN. The protocol is implemented in the OpenVMS device driver ESS $LASTDRIVER.

The LASTport Control Program (LASTCP) utility is the management interface that allows you to control and diagnose ESS$LASTDRIVER. You can use LASTCP to perform the following tasks:

- Start and stop ESS$LASTDRIVER
- Display counters for circuits, lines, nodes, and ESS$LASTDRIVER
- Display node characteristics
- Display known clients and servers
- Display LASTport status
- Reset counters

The description of the LASTCP utility covers the following topics:

- Invoking and exiting the utility
- LASTCP command summary
- Starting InfoServer Client for OpenVMS software automatically

11.5.1. Invoking and Exiting the LASTCP Utility

Use of LASTCP requires normal privileges, except where noted. To invoke LASTCP, enter the following command:

```
$ RUN SYS$SYSTEM:ESS$LASTCP
%LASTCP-I-VERSION, ESS$LASTDRIVER V1.5 is running
LASTCP>
```

At the LASTCP> prompt, you can enter LASTCP commands. To exit the utility, enter EXIT or press **Ctrl/Z** at the LASTCP> prompt.

You can also execute a single LASTCP command by using a DCL string assignment statement, as shown in the following example:

```
$ LASTCP :== $ESS$LASTCP
$ LASTCP SHOW CLIENTS
```

LASTCP executes the SHOW CLIENTS command and returns control to DCL command level.

11.5.2. LASTCP Command Summary

Table 11.2, "Summary of LASTCP Commands" summarizes LASTCP commands and their functions.

Table 11.2. Summary of LASTCP Commands

Command	Function
EXIT	Returns the user to DCL command level
HELP	Displays HELP text for LASTCP commands
SHOW CIRCUIT COUNTERS	Displays circuit counters
SHOW CLIENTS	Displays known clients
SHOW LINE COUNTERS	Displays line counters
SHOW NODE CHARACTERISTICS	Displays node characteristics
SHOW NODE COUNTERS	Displays node counters

Command	Function
SHOW SERVERS	Displays known servers
SHOW STATUS	Displays local status
SHOW TRANSPORT COUNTERS	Displays transport counters
START TRANSPORT	Starts LASTDRIVER
STOP TRANSPORT	Stops LASTDRIVER
ZERO COUNTERS	Resets counters

You can abbreviate LASTCP commands to the first unique characters of the command verb. For example, you can abbreviate the command SHOW SERVERS to SHSE.

LASTCP provides a Help facility that contains information about each command and its parameters and qualifiers, as well as examples of its use. For a complete description of LASTCP commands, refer to the *InfoServer Client for OpenVMS LASTCP and LADCP Utilities* manual.

11.5.3. Starting InfoServer Client for OpenVMS Software Automatically

You must start InfoServer Client for OpenVMS software using the ESS$STARTUP command procedure. To make sure the software is started automatically each time the system reboots, execute the startup procedure from within SYSTARTUP_VMS.COM.

How to Perform This Task

1. Determine the value of SCSNODE, your system's node name parameter. If the parameter is defined as the null string (the default value), InfoServerClient for OpenVMS software does not start.

   If you are running or plan to run DECnet for OpenVMS, SCSNODE must be defined as the system's DECnet node name. If you do not plan to run DECnet, and if the system is an OpenVMS cluster member, SCSNODE must be defined as the SCS system name, a 1- to 8-character node name that is unique in the cluster.

   To determine the value of SCSNODE, enter the following commands to invoke SYSMAN and display the parameter:

   ```
   $ RUN SYS$SYSTEM:SYSMAN
   SYSMAN> PARAMETERS USE CURRENT
   SYSMAN> PARAMETERS SHOW SCSNODE
   ```

2. If SCSNODE is defined as the null string, perform these steps:

   a. Enter a command in the following format, where *node-name* is the system's DECnet node name or (if you do not plan to run DECnet for OpenVMS) the SCS system name:

      ```
      PARAMETERS SET SCSNODE "node-name"
      ```

      For example:

      ```
      SYSMAN> PARAMETERS SET SCSNODE "MYNODE"
      ```

   b. Enter the following commands to write the new value to the parameter file and exit from SYSMAN:

```
SYSMAN> PARAMETERS WRITE CURRENT
SYSMAN> EXIT
```

c. Add a line in the following format to the AUTOGEN parameter file SYS $SYSTEM:MODPARAMS.DAT to define the SCSNODE parameter:

```
SCSNODE = "node-name"
```

For example:

```
SCSNODE = "MYNODE"
```

Note

The documented 6-character maximum size for SCSNODE is strictly enforced. The value of SCSNODE will be truncated by SYSBOOT if the size is set to anything over six characters in the system parameter file.

---

3. Invoke any editor to edit SYS$MANAGER: SYSTARTUP_VMS.COM and find the command that starts InfoServer Client software. For example:

```
$ @SYS$STARTUP:ESS$STARTUP DISK
```

Note that the parameters CLIENT and DISK are synonymous. If the command is preceded by the DCL comment delimiter (!), remove the delimiter. To enable tape functions, add the TAPE parameter to the command line:

```
$ @SYS$STARTUP:ESS$STARTUP DISK TAPE
```

4. If SYSTARTUP_VMS.COM invokes the DECnet for OpenVMS startup procedure (SYS $MANAGER:STARTNET.COM), make sure SYSTARTUP_VMS.COM executes the InfoServer Client for OpenVMS startup procedure *after* invoking STARTNET.COM.

The following example shows the network startup command line followed by the InfoServer Client for OpenVMS startup command line. Note that if you omit the TAPE parameter, only the disk function is started.

```
$ @SYS$MANAGER:STARTNET
.
.
.
$ @SYS$STARTUP:ESS$STARTUP DISK TAPE
```

5. Optionally, edit the file SYS$STARTUP: ESS$LAST_STARTUP.DAT to specify desired startup qualifiers for the LASTport transport. (Refer to the *InfoServer Client for OpenVMS LASTCP and LADCP Utilities* manual.)

11.5.4. InfoServer Client Can Fail to Start If DECnet Is Started or Stopped

The InfoServer client software fails to start on a system where DECnet has been started and subsequently stopped. The following message will be found in the file SYS$MANAGER:ESS $STARTUP.LOG:

```
%ESS-I-NONET ESS started before DECnet.  4-MAR-2016 16:36:39.29
```

If the InfoServer client must be started at this point, the LASTport transport can be started with the Last Control Program using the following command:

```
$ MCR ESS$LASTCP
LASTCP> START
```

This command will start the transport. You can now execute the InfoServer client startup:

```
$ @SYS$STARTUP:ESS$STARTUP DISK
```

Because the transport is already started, the startup will run successfully.

11.5.5. Multiple Controllers Configured But Not All Attached to Media (Alpha Only)

If you have multiple Ethernet and FDDI controllers configured on your OpenVMS Alpha system, you might experience problems with the InfoServer client transport (LASTport) under either of the following conditions:

- Not all of the Ethernet and FDDI controllers are connected to network cabling.
- An FDDI controller is connected to the network cabling, but the FDDI ring is not functional; for example, some FDDI hardware may be powered down or broken.

Problems can range from not being able to access all the services available on the network, if you have four or more controllers configured, to a system crash.

To avoid these problems, specify only the controllers that are attached to media. VSI recommends that you do this by first editing your SYS$STARTUP:ESS$LAST_STARTUP.DAT data file to specify only the controllers that are attached and then restarting your system.

With certain controller configurations, if you specify controllers that are not attached, your system might crash when you issue the following command sequence:

```
$ MC ESS$LASTCP
LASTCP> STOP
```

An example of how to edit the SYS$STARTUP:ESS$LAST_STARTUP.DAT file follows. The unedited file is shown first, followed by an edited file.

```
!++
! This file will be used to set the appropriate LASTCP qualifiers. The
 following
! LASTCP qualifiers: ALL_CONTROLLERS, CHECKSUM, TRANSMIT_QUOTA, or
 SLOW_MODE
! can be set by using the following statement format:
! LASTCP qualifier = 1 to enable   e.g. SLOW_MODE = 1 enables  SLOW_MODE
! LASTCP qualifier = 0 to disable  e.g. SLOW_MODE = 0 disables SLOW_MODE
! The remaining LASTCP qualifiers will require the appropriate value
 settings.
! DEVICE          = (list-of-devices)
! TIMEOUT    = n    minimum interval in seconds
! CIRCUIT_MAXIMUM = n    maximum number of nodes
! GROUP           = n    Group number
! NODE_NAME       = name   Node name
! CONTROLLERS     = ([{controller letter,}...]) Controller list
! TRANSMIT_QUOTA  = n    Number of transmit buffers
!--
```

```
ALL_CONTROLLERS = ON
```

The edited SYS$STARTUP:ESS$LAST_STARTUP.DAT file follows. This example assumes you have ESA, ETA, EXA, EZA controllers configured on your system and that only the ESA controller is attached to the Ethernet wire.

```
!++
! This file will be used to set the appropriate LASTCP qualifiers. The
 following
! LASTCP qualifiers: ALL_CONTROLLERS, CHECKSUM, TRANSMIT_QUOTA, or
 SLOW_MODE
! can be set by using the following statement format:
! LASTCP qualifier = 1 to enable   e.g. SLOW_MODE = 1 enables  SLOW_MODE
! LASTCP qualifier = 0 to disable  e.g. SLOW_MODE = 0 disables SLOW_MODE
! The remaining LASTCP qualifiers will require the appropriate value
 settings.
! DEVICE          = (list-of-devices)
! TIMEOUT    = n    minimum interval in seconds
! CIRCUIT_MAXIMUM = n    maximum number of nodes
! GROUP           = n    Group number
! NODE_NAME       = name   Node name
! CONTROLLERS     = ([{controller letter,}...]) Controller list
! TRANSMIT_QUOTA  = n    Number of transmit buffers
!--
ALL_CONTROLLERS = OFF
DEVICE = (ESA)
```

Note

The default ESS$LAST_STARTUP.DAT file is stored in SYS$COMMON:[SYS$STARTUP].You might want to put the edited file in SYS$SPECIFIC:[SYS$STARTUP]. Otherwise, other system roots might be affected.

11.5.6. Startup Restrictions: PATHWORKS and RSM

If PATHWORKS or Remote System Manager (RSM) or both are installed, the InfoServer Client for OpenVMS startup must be run before the startup for PATHWORKS or RSM, or both. For example:

```
$ @SYS$MANAGER:STARTNET
⋮
$ @SYS$STARTUP:ESS$STARTUP DISK TAPE
$ @SYS$STARTUP:PCFS_STARTUP
$ @SYS$STARTUP:RSM$SERVER_STARTUP
```

InfoServer Client for OpenVMS software provides device drivers and control programs that are shared by both the PATHWORKS and RSM products. All InfoServer Client for OpenVMS components are prefixed with ESS$. The drivers and control programs supplied with InfoServer Client for OpenVMS software provide all necessary support for both PATHWORKS and RSM in addition to InfoServer Client support. You must execute the InfoServer Client for OpenVMS startup in the site-specific startup before executing either the PATHWORKS or RSM startup procedure.

11.5.7. Startup Restrictions: SYSMAN

You cannot start InfoServer Client for OpenVMS from a subprocess. Because the OpenVMS System Management utility (SYSMAN) uses subprocesses to complete its tasks on remote nodes, SYSMAN cannot be used to execute the SYS$STARTUP:ESS$STARTUP procedure.

11.5.8. User Account Requirements

To work with InfoServer Client for OpenVMS software, user accounts on your system must have the following privileges and quotas:

- GRPNAM privilege to use the /GROUP qualifier of the LADCP command BIND; SYSNAM privilege to use the command's /SYSTEM qualifier.
- At a minimum, default UAF account quotas.

Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for an explanation of how to verify and change account privileges and quotas.

11.5.9. System Parameter MAXBUF Requirement

To use all the LASTport Control Program (LASTCP) utility's SHOW functions, you must set the value of the system parameter MAXBUF to 32000 or greater.

11.6. Understanding LADCP Utility Functions

Use the LAD Control Program (LADCP) utility to configure and control the LASTport/Disk and LASTport/Tape protocols on OpenVMS systems. OpenVMS systems that use LASTport/Disk and LASTport/Tape services are called client systems. You can use LADCP to perform the following tasks:

- Establish **bindings** to services. A binding creates a new DAD `n`: virtual disk unit or a new MAD `n`: virtual tape unit on the local OpenVMS system.
- Remove bindings to services.

You can control service access by using a service access password. You can also write-protect services. In this case, local OpenVMS users of a DAD `n`: or MAD `n`: device unit receive an error if they attempt a write operation to the unit.

The protocols allow you to access storage devices that reside on an InfoServer system as though they are locally connected to your OpenVMS system. Thus, several OpenVMS client systems can share the same read-only media, eliminating the need for duplicate drives and media.

DAD `n`: and MAD `n`: device units are also referred to as **virtual device units**. They represent the local OpenVMS context for a volume that resides on a remote server. The OpenVMS driver that controls the DAD `n`: units is called ESS$DADDRIVER. The OpenVMS driver that controls the MAD `n`: units is called ESS$MADDRIVER.

The LASTport/Disk and LASTport/Tape protocols depend on the LASTport transport. The ESS $STARTUP.COM command procedure in SYS$STARTUP automatically loads ESS$DADDRIVER and ESS$MADDRIVER as well as ESS$LASTDRIVER, the LASTport transport driver.

Note

Your site-specific startup command procedure must include a call to ESS$STARTUP.COM. If you are using DECnet software, you must place the call *after* the @SYS$MANAGER:STARTNET.COM command that starts DECnet software. See *Section 11.5.3, "Starting InfoServer Client for OpenVMS Software Automatically"*.

11.6.1. Invoking and Exiting the LADCP Utility

To invoke LADCP, enter the following command:

```
$ RUN SYS$SYSTEM:ESS$LADCP
LADCP>
```

You can enter LADCP commands at the LADCP> prompt.

You can also execute a single LADCP command by using a DCL string assignment statement, as shown in the following example:

```
$ LADCP :== $ESS$LADCP
$ LADCP BIND CD_DOC_00661 /NOWRITE
```

LADCP executes the BIND command and returns control to DCL command level.

To exit LADCP, enter EXIT or press **Ctrl/Z** after the LADCP> prompt.

11.6.2. LADCP Command Summary

Table 11.3, "Summary of LADCP Commands" summarizes LADCP commands and their functions.

Table 11.3. Summary of LADCP Commands

Command	Function
BIND	Establishes a service binding and creates a device unit
DEALLOCATE	Terminates any active connection to a service without deleting the unit control block (UCB)
EXIT	Returns the user to DCL command level
HELP	Displays help text for LADCP commands
SHOW SERVICES	Displays services offered by InfoServer systems on the LAN
UNBIND	Terminates an established service binding

LADCP provides a Help facility that contains information about each LADCP command, including parameters, qualifiers, and examples of its use. For detailed descriptions of LADCP commands, refer to the *InfoServer Client for OpenVMS LASTCP and LADCP Utilities* manual.

11.6.3. Making InfoServer Devices Available Automatically

You can make remote InfoServer devices available on your system each time the system boots. To do so, add to SYSTARTUP_VMS.COM a series of LADCP BIND commands. For more information about the BIND command, refer to the *InfoServer Client for OpenVMS LASTCP and LADCP Utilities* manual.

How to Perform This Task

1. Edit SYSTARTUP_VMS.COM and find the command that starts InfoServerClient software. For example:

   ```
   @SYS$STARTUP:ESS$STARTUP DISK TAPE
   ```

This command starts the software with disk and tape functions.

2. Add the following command to invoke LADCP:

   ```
   $ RUN SYS$SYSTEM:ESS$LADCP
   ```

3. Immediately after this command, add BIND commands in the following format to make InfoServer compact discs or read/write disks available as virtual device units:

   ```
   BIND [/QUALIFIER,...] service-name
   ```

   To make tape devices available, you must specify the /TAPE qualifier in addition to any other desired qualifiers:

   ```
   BIND/TAPE [/QUALIFIER,...] service-name
   ```

   For `service-name`, specify the name of the InfoServer device service. Usually a service name is the label of the volume to which the InfoServer system is providing access. For more information about the BIND command, refer to the *InfoServer Client for OpenVMS LASTCP and LADCP Utilities* manual.

4. Add an EXIT command to exit LADCP.

5. Add MOUNT commands in the following format to make available as public devices the virtual device units created in the previous step:

   ```
   MOUNT/SYSTEM/NOASSIST device-name volume-label
   ```

   For `device-name`, specify the name of the device. For `volume-label`, specify a volume label to assign to the device. For more information about the MOUNT command, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

Example

The following commands, executed in SYSTARTUP_VMS.COM, start the InfoServerClient software and make available to client systems the InfoServer device DAD$OPENVMSV72.

```
⋮
$ @SYS$STARTUP:ESS$STARTUP DISK
$ RUN SYS$SYSTEM:ESS$LADCP
  BIND OPENVMSV72
  EXIT
$ MOUNT/SYSTEM/NOASSIST DAD$VMS055 VMS055
⋮
```

In this example, the OpenVMS Version 7.2 consolidated distribution (CONdist) compact disc loaded in a compact disc drive connected to an InfoServer system, is made available on the server as a virtual device unit and mounted as a public device.

Chapter 12. Managing the LAT Software

This chapter describes how the LAT software works and the tasks you must perform to implement and manage the LAT software on your system.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Starting up the LAT protocol	*Section 12.5, "Starting Up the LAT Protocol"*
Customizing LAT characteristics	*Section 12.6, "Customizing LAT Characteristics"*
Creating a service	*Section 12.6.1, "Creating Additional Services"*
Setting up ports	*Section 12.6.2, "Setting Up Ports"*
Setting up printers	*Section 12.6.2.1, "Setting Up Printers"*
Setting up special application services	*Section 12.6.2.2, "Setting Up Special Application Services"*
Enabling queued incoming requests	*Section 12.6.3, "Queuing Incoming Requests "*
Enabling outgoing LAT connections	*Section 12.6.4, "Enabling Outgoing LAT Connections"*
Managing the LATACP database size	*Section 12.7, "Managing the LATACP Database Size"*

This chapter explains the following concepts:

Concept	Section
LAT protocol	*Section 12.1, "Understanding the LAT Protocol"*

Concept	Section
LAT network	*Section 12.2, "Understanding the LAT Network"*
LAT configurations	*Section 12.3, "Understanding LAT Configurations"*
LAT Control Program utility	*Section 12.4, "Understanding the LATCP Utility"*

12.1. Understanding the LAT Protocol

The operating system uses the LAT (local area transport) software to communicate with **terminal servers** and other systems within a local area network (LAN). Terminal servers are communication devices dedicated for connecting terminals, modems, or printers to a LAN. They offer the following features:

- Provide a cost-effective method of connecting many user terminals to a computer
- Save on cable requirements
- Maximize the number of devices that can access a computer

With the LAT software, which implements the LAT **protocol**, the operating system can offer resources, or **services**, that the terminal servers can access. A system that offers LAT services is called a **service node**. In addition, nodes can access LAT services by enabling outgoing connections (using LATCP) and using the DCL command SET HOST/LAT. (In the remainder of this chapter, “servers” refers both to dedicated terminal servers and to nodes that allow outgoing access to other LAT services.)

A LAT service can consist of all the resources of a computer system, or it can be a specific resource on a computer system, such as an application program. You can set up your system as a **general time-sharing service**, meaning that all of its resources are available to users in the LAN, or you can restrict access to a specific service (application program) on the system. This chapter and the *VSI OpenVMS I/O User's Reference Manual* outline the procedure you use to set up access to a dedicated application program.

12.1.1. How the LAT Protocol Works

The LAT protocol allows the terminal servers and computers to communicate within a LAN, such as the Ethernet or the Fiber Distributed Data Interconnect (FDDI). The LAT protocol matches terminals and other devices to the computing resources (services) of the LAN. Because LAT terminals are not connected directly to the computer (service node) they are accessing, the local server must listen for service requests from its terminals and be able to match the terminals with computers that provide the desired services.

Using the LAT protocol, then, the operating system announces its available services over the LAN. Servers listen to the LAN announcements and build a database of service information so that they can locate an appropriate system when a user terminal requests computing services. For example, a user terminal might request general processing service or a data entry program on the operating system. A server uses the LAT protocol to establish and maintain a connection between the requesting terminal and the operating system.

Sometimes the operating system can request services from a terminal server. The LAT protocol allows systems to ask for connections to printers or other devices attached to a terminal server.

12.1.2. Advantages of the LAT Protocol

Using the LAT protocol on your system has many advantages:

- The LAT protocol lets you make the resources of any computer on a local area network available to any user in that network.
- In addition to general processing resources, you can set up terminals, printers, and modems so they are available from multiple systems in the local area network. This lets you efficiently use these resources and keep them available even if one of the systems in the network must be shut down.
- You can also set up application programs, such as data entry programs or news services, as resources. When a user requests a connection to the resource, the LAT protocol sets up a connection directly to the application program. No login procedure is necessary.
- The LAT protocol provides load balancing features and recovery mechanisms so users get the best, most consistent service possible. In their broadcast messages, systems rate the availability of their services so that servers can establish connections to computing resources on the least busy node. If a node becomes unavailable for any reason, the servers attempt to provide access to alternate services.
- Users can establish multiple computing sessions on their terminals, connecting to several different computers and switching easily from one computing session to another. After switching from one session to another, users can return to the previous session and pick up where they left off. This saves users the time normally required to close out and reopen files or accounts and to return to the same point in a session.
- Finally, the LAT protocol can provide improved system performance. Because the servers bundle messages onto a single LAN interface, a server interface decreases the network traffic and reduces the number of computer interrupts realized in systems where terminals, modems, and printers each have a physical connection to the computer.

12.2. Understanding the LAT Network

A **LAT network** is any local area network where terminal servers and operating systems use the LAT protocol. A LAT network can coexist on the same LAN with other protocols. The LAT protocol, which operates on both terminal servers and the operating systems, is designed to ensure the safe transmission of data over the LAN.

The LAT network consists of the following components:

Component	**For More Information**
Service nodes	*Section 12.2.1, "Service Nodes"*
Terminal server nodes	*Section 12.2.2, "Terminal Server Nodes"*
Nodes allowing outgoing connections	*Section 12.2.3, "Nodes Allowing Outgoing Connections"*

Component	For More Information
LAN cable	*Section 12.2.4, "Components of a LAT Network"*

Service nodes supply computing resources for the local network, while terminal server nodes (or nodes allowing outgoing connections) port their terminals, modems, or printers to those resources upon request from a user terminal or an application program.

Note that in a LAT network, nodes that *access* services are often referred to as *master* nodes, which distinguishes them from nodes that only *provide* services.

You can use the LAT Control Program (LATCP) to configure the LAT characteristics for your system. LATCP allows you to set up your system to support:

- Incoming access only
- Outgoing access only
- Both incoming and outgoing access

The systems that support incoming LAT connections are **service nodes**. (Using LATCP, you can also set up your system so that it supports neither incoming nor outgoing access.)

12.2.1. Service Nodes

A service node is one type of node in a LAT network. (Nodes that are not running an OpenVMS operating system can also be used along with the OpenVMS nodes in a LAT network.) A service node is an individual computer in a LAN that offers its resources to users and devices. Because the OpenVMS operating systems contain the LAT protocol, any OpenVMS system can be configured as a service node within a LAT network.

12.2.1.1. Types of Services

Each node offers its resources as a **service**. Often, a node offers a general processing service, but it can offer limited services or special application services as well. Any or all of the services can be specialized applications.

For example, your service node might offer services for the following items:

- General processing
- Data entry
- Stock quotations

The general processing service would allow the use of the general computing environment. The data entry and stock services, on the other hand, would be restricted environments, with connections to the application service but to no other part of the service node.

Each service is distinguished by the name the system manager assigns to it. In an OpenVMS Cluster, VSI recommends that the service name be the same as the cluster name. In an independent node, VSI recommends that the service name be the same as the node name. With special service applications, the service holds the name of the application.

12.2.1.2. Service Announcements

A service node announces its services over the LAN at regular intervals so that terminal servers (and OpenVMS systems that allow outgoing connections) know about the availability of these network resources. The service announcement provides the physical node name, the service names, a description of services, and a rating of service availability. Servers listen to the LAN announcements and record information in a database. On nodes allowing outgoing connections, this database is maintained by the LAT Ancillary Control Process (LATACP). (See *Section 12.7, "Managing the LATACP Database Size"* for more information about managing the LATACP database.)

Whenever a user terminal or application program requests a service, the server node connects to the appropriate service node.

Note that you can disable a local node from multicasting service announcements by using the /NOANNOUNCEMENTS qualifier with the LATCP command SET NODE. However, because remote nodes must rely on the LAT service responder feature in the LAT protocol Version 5.2 (or higher) to connect to the local node, VSI recommends that you use this qualifier only in a networking environment where newer model terminal servers and hosts are present (all LAT hosts, terminal servers, and PCs are running at least Version 5.2 of the LAT protocol). Otherwise, systems running versions of the LAT protocol prior to Version 5.2 (for example, DECserver 100, 200, and 500 systems) will be unable to connect to any of the systems that have LAT service announcements disabled.)

12.2.1.3. Print Requests

In some cases, service nodes can request services from terminal servers. The most common situation is when the system wants to use a printer that is connected to a terminal server port. The system submits the print request to the terminal server print queue that is set up and initialized in the OpenVMS startup procedure. Then the LAT symbiont (the process that transfers data to or from mass storage devices) requests the LAT port driver to create and terminate connections to the remote printer.

For information about setting up queues for printers connected to LAT ports, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

12.2.2. Terminal Server Nodes

A **terminal server node** is the second type of node in a LAT network. A terminal server node is usually located near the terminals and printers it supports. The terminals and printers are physically cabled to the terminal server; the terminal server is physically connected to the LAN cable.

12.2.2.1. Locating Service Nodes

Terminal servers build and maintain a directory of services from announcements advertised over the network. Then, when terminal servers receive requests from terminal users, they can scan their service databases and locate the computer that offers the requested service.

Terminal servers not only look for the node that provides the requested service, but they can also evaluate the service rating of that node. If a requested service is offered by more than one node, then the service rating is used to select the node that is least busy. A server establishes a logical connection between the user terminal and the service node.

12.2.2.2. Setting Up Connections

One logical connection carries all the data directed from one terminal server node to a service node. That is, the server combines data from all terminals communicating with the same node onto one

connection. A terminal server establishes a logical connection with a service node only if a logical connection does not already exist.

If a connection fails for any reason, a terminal server attempts to find another node offering the same service and “rolls over” the connection so users can continue their computing sessions.

Even though terminal connections are bundled together, each terminal can be uniquely identified by its name. A terminal name consists of two parts: the first part is the name of the port on the terminal server that the terminal line plugs into; the second part is the name of the terminal server node.

12.2.2.3. Servicing Nodes

Although terminal servers are usually the requesting nodes in a LAT network, sometimes service nodes request service from terminal servers. Most commonly, a service node queues print requests to remote printers connected to terminal servers.

12.2.3. Nodes Allowing Outgoing Connections

Nodes can be set up to allow incoming connections, outgoing connections, or both. Nodes (excluding those that offer incoming connections only) such as terminal server nodes can locate service nodes and set up connections. The database of information about available nodes and services is maintained by the LAT Ancillary Control Process (LATACP). (See *Section 12.7, "Managing the LATACP Database Size"* for more information about managing the LATACP database.)

On a node that is set up to allow outgoing LAT connections, a user can connect to another node in the LAT network by entering the SET HOST/LAT command. For more information, refer to the SET HOST/LAT command in the *VSI OpenVMS DCL Dictionary*.

12.2.4. Components of a LAT Network

Figure 12.1, "Components of a LAT Network" is an example of a LAT network. The network consists of an Ethernet cable connecting service nodes and terminal server nodes.

The three service nodes in *Figure 12.1, "Components of a LAT Network"*, named MOE, LARRY, and ALEXIS, each offer services to terminal server nodes on the network.

Two of the service nodes, MOE and LARRY, belong to the OFFICE cluster. (The cluster is distinguished by its computer interconnect [CI] and star coupler.) Because MOE and LARRY are clustered, their service names are the same as their cluster name. Because both service nodes offer an OFFICE service, terminal server nodes can assess the work load on both OFFICE nodes and establish a connection to a node that offers the service that is less busy.

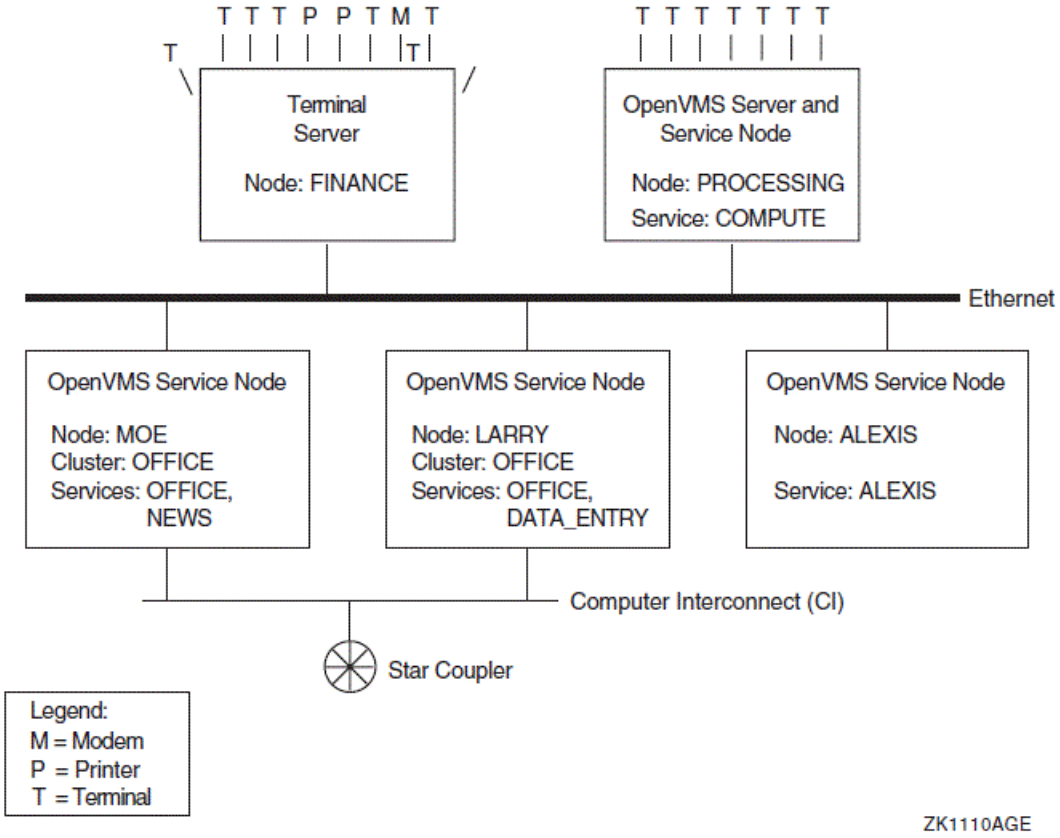
The third service node, ALEXIS, is an independent node in the LAT network so its service name is the same as its node name.

In addition to its primary OFFICE service, node MOE offers an application service called NEWS. With this specialized service, user terminals can connect directly to the online news program, without any login procedure but also without general access to the general computer resources of the node.

The node FINANCE, shown in *Figure 12.1, "Components of a LAT Network"*, is a terminal server node; it supports a number of interactive terminals, a modem, and printers. The node PROCESSING allows outgoing connections; it supports several interactive terminals. The node FINANCE can accept print requests from any of the three service nodes, provided each service node has set up print queues to support remote printers on the terminal server.

Node PROCESSING is also a service node. It offers the service COMPUTE.

Figure 12.1. Components of a LAT Network



12.3. Understanding LAT Configurations

When you set up a LAT system, you need to understand the relationship between the LAT software and the network so you can configure your system to operate efficiently. The following sections contain information that will help you understand the following concepts:

- The relationship between LAT software and OpenVMS Cluster software
- The relationship between LAT software and DECnet software
- How the LAT software works in a networking environment that uses multiple LAN adapters
- How to use the LAT software in an Ethernet/FDDI configuration that cannot use large buffers

12.3.1. LAT Relationship to OpenVMS Clusters and DECnet

Although the LAT protocol works independently of OpenVMS Cluster software, VSI recommends that you configure a service node to complement the OpenVMS Cluster concept. You achieve this by creating a service on each node in an OpenVMS Cluster and assigning the cluster name to this service. A terminal server assesses the availability of cluster services and establishes a connection to the node that is least busy. Thus, the LAT protocol helps balance the cluster load. If one node in the cluster fails, the terminal server can transfer the failed connections to another service node within the cluster.

The LAT software does not use DECnet as a message transport facility, but instead uses its own virtual circuit layer to implement a transport mechanism. The LAT and DECnet software work independently in a common LAN environment. For compatibility, if a service node is also a DECnet node, the service node name should be the same as the DECnet node name.

12.3.1.1. LAT and DECnet Running on the Same Controller

If Ethernet ports will be running both DECnet and LAT, you must start the DECnet software *before* the LAT software. If you do not start DECnet software first, all existing LAT connections may terminate, and reconnecting to the system via LAT may not be possible.

12.3.1.2. LAT and DECnet Running on Different Controllers

If DECnet is configured on the system (or if the system is part of a cluster), the SCSSYSTEMID system parameter may contain a nonzero value. Normally, this is not a problem unless the system has two or more LAN controllers connected to the same logical LAN.

For example, if your system has an FDDI controller and an Ethernet controller, your site may be configured so that the FDDI ring attached to the FDDI controller and the Ethernet segment attached to the Ethernet controller are bridged by a 10/100 LAN bridge (FDDI-to-Ethernet). In this configuration, it is impossible to run LAT over both controllers.

In such a configuration, you *must* run LAT and DECnet over the same controller if SCSSYSTEMID is not 0. If they do not run on the same controller, DECnet starts first, which in turn causes the LAT startup on the other controller to fail. This failure occurs because LAT startup tries to use the AA-00-04-00-xx-xx address (the DECnet LAN address); however, because DECnet is already using this address on another controller, the data link layer prevents the LAT startup from using that address. (In a single logical LAN, all data link addresses must be unique. Because both controllers try to use the same address, it is no longer unique.)

Using the following command to create the LAT link also fails because the LAN driver tries to use the address based on SCSSYSTEMID:

```
LATCP> CREATE LINK LAT$LINK_2 /NODECNET
```

If SCSSYSTEMID is set to 0, configuring LAT and DECnet on different controllers is possible. However, in a cluster environment, SCSSYSTEMID cannot be set to 0.

12.3.2. Using Multiple LAN Adapters

When you use multiple LAN addresses for one LAT node, you can configure a system with multiple LAN adapters connected to the same logical LAN. The LAT software can run over each adapter simultaneously and can better maintain connections. For example, when a virtual circuit chooses a primary path and uses it for all LAT message transmissions, the LAT software can continue communications through another adapter or logical path if that original path becomes blocked.

Note

Nodes running versions of LAT software prior to Version 5.3 of the LAT protocol (included in the OpenVMS operating system beginning with Version 7.0) may exhibit some differences in behavior. Therefore, if your configuration includes earlier versions of the LAT software, such as Version 5.1 or Version 5.2, note the differences and considerations discussed in this chapter.

12.3.2.1. Supported Configurations

Although it is possible to run LAT over multiple LAN adapters, it is still not possible to route LAT from one logical LAN to another. The following examples show supported LAT configurations for nodes running Version 5.3 of the LAT protocol (including nodes running Version 5.2 and 5.1 as well).

The widely used configuration shown in *Figure 12.2, "Multiple Address LAT Configuration: One LAN with Mixed-Version LAT Nodes"* has an OpenVMS system running LAT Version 5.3 software over two Ethernet adapters (labeled A and B in the diagram) connected to the same physical LAN as a DECserver 200.

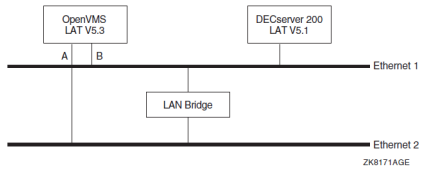
Figure 12.2. Multiple Address LAT Configuration: One LAN with Mixed-Version LAT Nodes

When a LAT connection is started between the DECserver 200 and the OpenVMS system, the LAT software determines that it is possible to use both adapters A and B for the LAT virtual circuit. One of the adapters will be chosen as the primary communications path while the other will be present in the unlikely event that the primary path fails.

For example, if a user connects to the OpenVMS system from the DECserver 200, the OpenVMS system determines that there are two paths but chooses adapter B as the primary communications path. If the user runs a program that generates a large amount of output from the OpenVMS system and adapter B fails in some manner during the output, the LAT software will attempt to continue communications from the OpenVMS system to the DECserver through adapter A.

Figure 12.3, "Multiple Address LAT Configuration: Two LANs with Mixed Version LAT Nodes" shows two LANs bridged together. However, this configuration will have the same characteristics as the configuration shown in *Figure 12.2, "Multiple Address LAT Configuration: One LAN with Mixed-Version LAT Nodes"*.

Figure 12.3. Multiple Address LAT Configuration: Two LANs with Mixed Version LAT Nodes

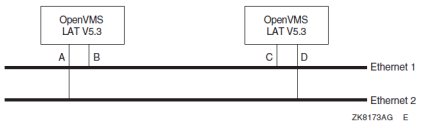


Note

It is possible for Ethernet 2 in *Figure 12.3, "Multiple Address LAT Configuration: Two LANs with Mixed Version LAT Nodes"* to be an FDDI network. The LAT software regards each adapter as a network path with equal point-to-point communications and does not treat FDDI controllers any differently. However, for large buffer support, see *Section 12.3.3, "Large Buffers in Ethernet/FDDI Configurations"* for more details.

In *Figure 12.4, "Multiple Address LAT Configuration: Two LANs with Version 5.3 LAT Nodes"*, any virtual circuit created between the two OpenVMS systems will have two paths: through controllers B and C or A and D. If one path fails, the virtual circuit will continue over the other path. If both paths fail, the virtual circuit will eventually time out.

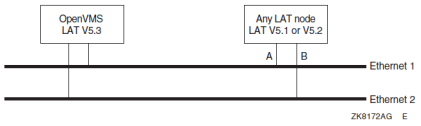
Figure 12.4. Multiple Address LAT Configuration: Two LANs with Version 5.3 LAT Nodes



12.3.2.2. Unsupported Configuration

When configuring a network to use an OpenVMS system running the LAT Version 5.3 software, *avoid* the configuration shown in *Figure 12.5, "Unsupported Multiple Address LAT Configuration"*.

Figure 12.5. Unsupported Multiple Address LAT Configuration



Any configuration similar to this diagram will result in unpredictable results and may not function. In a network environment, LAT Version 5.1 and 5.2 nodes can have only a single logical LAN address. The configuration in *Figure 12.5, "Unsupported Multiple Address LAT Configuration"* violates this rule. The configuration shown in *Figure 12.4, "Multiple Address LAT Configuration: Two LANs with Version 5.3 LAT Nodes"* is a valid alternative.

12.3.2.3. Creating Logical LAT Links

The LAT software regards all paths as equal, point-to-point communications. The LAT software can support a maximum of eight LAN adapters simultaneously (and it is possible to connect all controllers to the same logical LAN). To get the maximum coverage over possible path failures, each logical link should be created prior to setting the LAT node state to ON in SYS$MANAGER:LAT $SYSTARTUP.COM.

For example, if a system has one Ethernet adapter (device ESA0) with two FDDI adapters (FCA0 and FCB0) and the system manager chooses to run LAT over all adapters, the LAT$SYSTARTUP.COM file would contain the following commands:

```
$!
$! Create each logical LAT link with a unique name and
$! unique LAN address (forced with /NODECNET).
$!
$ LCP CREATE LINK ETHERNET /DEVICE=ESA0 /NODECNET
$ LCP CREATE LINK FDDI_1 /DEVICE=FCA0 /NODECNET
$ LCP CREATE LINK FDDI_2 /DEVICE=FCB0 /NODECNET
$!
$! Turn on the LAT protocol.
$!
$ LCP SET NODE /STATE=ON
```

Caution

If the LATCP command SET NODE /STATE=ON is entered before the link is created, a random or default LAT$LINK will be created on one of the LAN adapters. There is no way to predict which LAN adapter will be chosen (it is dependent on the system configuration). Therefore, all logical LAT links should be created before LAT is started.

Be sure each logical link is created with the /NODECNET qualifier. It will prevent the possibility of link creation failure if multiple adapters attempt to use the DECnet style address. Having more than one LAN adapter connected to the same logical LAN with the same address violates LAN conventions and will cause problems with LAT and other protocols.

It is possible to create logical LAT datalinks after the LAT protocol has been started. Any existing virtual circuit will attempt to find any new paths through the newly created logical datalink when it is ready for use. However, VSI does not recommend that you create links at this point because during the time it takes existing virtual circuits to discover new paths through this newly created datalink, the virtual circuit may fail before the new paths are discovered.

12.3.2.4. Path Discovery

The OpenVMS LAT software uses a combination of the directory service and solicitation to obtain paths for each virtual circuit. To expedite path discovery at virtual circuit startup, VSI recommends that you configure a system with multiple LAN adapters to maintain a LAT service and node database, by performing the following actions:

- Enabling outgoing LAT connections
- Using the same group code mask for User Groups and Service Groups

An OpenVMS system running with outgoing connections disabled and no service and node database is still capable of running with multiple paths for each virtual circuit. These paths must be discovered through the LAT solicitation process and will take longer (leaving the possibility for virtual circuit failure to occur before all paths have been discovered).

12.3.2.5. Modifying LAT Parameters

In the unlikely event of a path failure, it will take the OpenVMS LAT software time (which will vary depending on the number of adapters to which the remote node has access) to locate another working path. Therefore, VSI strongly recommends that you modify the following LAT parameters on potential LAT master nodes:

- Retransmit limit - default value is 8. Set to the maximum number of LAN adapters times 8. For example, if an OpenVMS system on the LAN has 3 adapters, each LAT master node should have their retransmit limit set to 24 (3 * 8).
- Keepalive timer - default value is 20 seconds. While the default value may be sufficient in most circumstances, it may be necessary to increase this to 30 or 40 seconds.

Although it is possible to keep virtual circuits running through multiple adapters to LAT Version 5.1 or LAT Version 5.2 master nodes, there is still a possibility that the connections to these nodes may fail.

LAT Version 5.2 and LAT Version 5.1 master nodes do not have the ability to recognize multiple paths to LAT nodes that provide services. They can only communicate with such nodes through one remote address at a time. Therefore, if a LAN path failure occurs when a LAT master node running LAT Version 5.1 or Version 5.2 attempts to connect to a remote LAT Version 5.3 node providing services, the LAT Version 5.3 node might not discover this failure in time and the LAT master node may time out the connection. You can partially solve this problem by increasing the retransmit limit to as high a setting as possible.

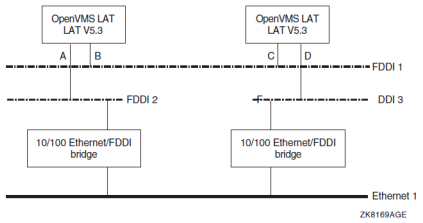
In addition, if a LAT Version 5.3 node providing services views the virtual circuit as completely idle during the primary path failure, no attempt will be made to use any of the alternate paths (because of the previously described LAT Version 5.2 and 5.1 limitation). Therefore, although multiple LAN adapters

will work with older LAT implementations, you might still need to upgrade to Version 7.0 or higher of the OpenVMS operating system to obtain the LAT Version 5.3 protocol, which will correct this type of problem. (Note that this type of problem affects only those connections that are idle. An example of where this situation could arise is in an office environment if all users were to leave their systems at the same time, either at lunchtime or at the end of the workday.)

12.3.3. Large Buffers in Ethernet/FDDI Configurations

The OpenVMS LAT software will attempt to use *large* buffers over any virtual circuit that comes in over an FDDI controller. This feature can cause problems if an alternate virtual circuit path must go through an Ethernet. *Figure 12.6, "LAT FDDI Ring and Large Buffers"* is an example of the configuration that can cause problems.

Figure 12.6. LAT FDDI Ring and Large Buffers



In this diagram, it is possible for the two OpenVMS systems to communicate using large packets through the path described by controllers B and C. *Large* packets may exceed 1500 bytes of data (the maximum Ethernet message can contain 1500 bytes of data). If the path described by controllers B and C were to fail, it will not be possible for communication to continue through the path described by A and D.

The path described by controllers A and D pass through an Ethernet LAN segment. The messages that are routed through the 10/100 bridges cannot be larger than the maximum Ethernet message. Problems can occur because the OpenVMS LAT software cannot always detect this kind of configuration.

There are two ways to prevent problems with the previously described configuration. The first and easiest option is to create a logical LAT link using an Ethernet adapter (if either system has an Ethernet LAN adapter). This will force the message size negotiation to be no larger than the maximum sized Ethernet message.

If neither system has an Ethernet controller (thus making the first option not possible), the second option is to override the use of large buffer support (which is enabled by default) by using the new LATCP command qualifier, /[NO]LARGE_BUFFER. For example:

```
$ MCR LATCP SET NODE/NOLARGE_BUFFER
```

VSI recommends that you use the SET NODE/NOLARGE_BUFFER command after all logical LAT links have been created and before the LAT node has been turned on. For example, note the order of the commands in LAT$SYSTARTUP.COM:

```
$!
$! Create each logical LAT link with a unique name and
$! unique LAN address (forced with /NODECNET).
$!
$ LCP CREATE LINK FDDI_1 /DEVICE=FCA0 /NODECNET
$ LCP CREATE LINK FDDI_2 /DEVICE=FCB0 /NODECNET
$!
$! Don't use large buffer support (force packet
$! sizes to be no larger than what Ethernet can
```

```
$! support).
$!
$ LCP SET NODE /NOLARGE_BUFFER
$!
$! Turn on the LAT protocol.
$!
$ LCP SET NODE /STATE=ON
```

12.4. Understanding the LATCP Utility

The LAT Control Program (LATCP) utility configures and controls the LAT software on OpenVMS systems. LATCP commands let you stop and start the LAT driver (which implements the LAT protocol) and modify or display LAT characteristics of the OpenVMS node.

With LATCP, you can set up your system as a service node, which offers one or more resources (services) for access by users on other systems in the local area network (LAN).

You can also use LATCP to set up the system to allow its users to access services on other systems in the LAN. In this case, the system can act like a terminal server: it can manage multiple user sessions simultaneously for connections to services on other nodes.

You can use LATCP to set up your system to support incoming access only, outgoing access only, or both incoming and outgoing access. You can also set up your system so that it supports neither incoming nor outgoing access.

When you set up your system to support outgoing access, the LAT software manages a database of LAT services and nodes. The software builds the database when you enable outgoing access on your node. The software begins to collect LAT **service announcements**—multicast messages sent by LAT service nodes—and builds the database based on these service announcements. You can use LATCP to display the services and nodes in this database and to control the size of the database. Allow outgoing access on systems that can tolerate the additional overhead, such as standalone systems.

Use LATCP to perform the following actions:

- Specify operational characteristics for your node and its services
- Turn the state of the LAT port driver (LTDRIVER) on and off
- Display the status of LAT services and service nodes in the network
- Display the status of links created on your LAT node
- Display the status of your LAT node
- Show and zero LAT counters
- Create, delete, and manage LAT ports
- Recall previously entered LATCP commands so that you can execute them again without having to reenter them
- Create subprocesses so that you can execute DCL commands without exiting from LATCP

With the LAT protocol, you can set up LAT application ports on the local node so that users can access printers and other asynchronous devices that are connected to LAT terminal servers or service nodes on the LAN. The remote devices must be configured appropriately.

12.4.1. Invoking and Exiting LATCP

Enter the following command to invoke LATCP:

```
$ RUN SYS$SYSTEM:LATCP
LATCP>
```

At the LATCP> prompt, you can enter LATCP commands. To exit LATCP, enter EXIT or press **Ctrl/Z** at the LATCP> prompt.

You can also execute a single LATCP command by using a DCL string assignment statement, as shown in the following example:

```
$ LCP :== $LATCP
$ LCP SET NODE/STATE=ON
```

LATCP executes the SET NODE command and returns control to DCL.

12.4.2. LATCP Commands

Table 12.1, "LATCP Commands" summarizes the LATCP commands.

Table 12.1. LATCP Commands

Command	Function
ATTACH	Transfers control from your current process to the specified process.
CREATE LINK	Creates LAT data links.
CREATE PORT	Creates a logical port on the local node.
CREATE SERVICE	Creates a service on a service node.
DEFINE/KEY	Assigns a command string to a function key on your keypad.
DELETE LINK	Deletes a LAT data link from a node.
DELETE PORT	Deletes an application port or dedicated port.
DELETE QUEUE_ENTRY	Deletes an incoming queued request from the local node.
DELETE SERVICE	Deletes a service on a service node.
EXIT	Returns the user to DCL command level.
HELP	Displays help text for LATCP commands.
RECALL	Recalls LATCP commands that you entered previously so that you can execute them again.
REFRESH	Refreshes your display screen, for example, after your display has been overwritten by output from some other source.
SCROLL	Allows you to retrieve information that has scrolled off the screen.
SET LINK	Modifies characteristics of LAT data links.
SET NODE	Specifies LAT characteristics for a node.
SET PORT	Maps a logical port on a node to either a remote device on a terminal server or a special application service on a remote LAT service node.
SET SERVICE	Changes service characteristics.
SHOW LINK	Displays the characteristics of links on your node.

Command	Function
SHOW NODE	Displays the characteristics of nodes.
SHOW PORT	Displays port characteristics.
SHOW QUEUE_ENTRY	Displays information about requests, or entries, queued on the local node.
SHOW SERVICE	Displays characteristics of LAT services known to your node.
SPAWN	Creates a subprocess.
ZERO COUNTERS	Resets the node counters, service counters, and link counters maintained by your node.

For detailed information about LATCP commands and qualifiers, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

12.5. Starting Up the LAT Protocol

As system manager, you start up the LAT protocol and configure your node as a service node by executing the command procedure SYS$STARTUP:LAT$STARTUP. This procedure executes the following two procedures:

1. LAT$CONFIG.COM, to load the LAT terminal driver LTDRIVER and create the LATACP process
2. LAT$SYSTARTUP.COM, to execute LATCP commands that define LAT characteristics

How to Perform This Task

To make sure the LAT protocol is started each time the system boots, add a command to execute this procedure in the general-purpose, site-specific startup command procedure, described as follows. (See *VSI OpenVMS System Manager's Manual, Volume 1: Essentials* for more detailed information about this command procedure, including the file specification used to identify it in your operating system.)

To set up your node as a LAT service node and start the LAT protocol software on your system each time the system boots, edit the general-purpose, site-specific startup command procedure to add the following line:

```
$ @SYS$STARTUP:LAT$STARTUP.COM
```

When the general-purpose, site-specific startup command procedure executes this command, it invokes LAT$STARTUP.COM, which in turn invokes the LAT$CONFIG and LAT$SYSTARTUP command procedures.

You can append any of the following arguments to the command line that invokes LAT$STARTUP to specify unique LAT characteristics for your node. The procedure will pass these arguments to LAT $SYSTARTUP.COM to define the LAT characteristics you specify.

```
$ @SYS$STARTUP:LAT$STARTUP "P1" "P2" "P3" "P4" "P5"
```

VSI recommends that you modify LAT$SYSTARTUP.COM directly, rather than passing parameters in `P1` through `P5`. However, if you choose to use P1 through P5, the arguments have the following meanings:

Argument	Format	Meaning
`P1`	Service-name	Name of the service. For clustered service nodes, use the cluster alias as the service name. For independent service nodes, use the DECnet node name. LAT$SYSTARTUP.COM

Argument	Format	Meaning
		uses the argument P1 to assign a service name to the node (with the LATCP CREATE SERVICE command).
P2–P4	Any of the following names:	LAT$SYSTARTUP.COM uses the arguments to assign LAT node characteristics (with the LATCP SET NODE command).

	/IDENTIFICATION= "string"	Description of the node and its services that is advertised over the local area network (LAN). The default is the string defined by the logical name SYS$ANNOUNCE. Make sure you include five sets of quotation marks around the identification string. For example:
		`"/IDENTIFICATION=" -`
		`  """""Official system center"""""`

	/GROUPS=(ENABLE= group-list)	Terminal server groups qualified to establish connections with the service node. By default, group 0 is enabled.
	/GROUPS=(DISABLE= group-list)	Removes previously enabled terminal server groups. If you are specifying the preceding qualifier to enable groups, you can combine the qualifiers into one, as shown in the example that follows this table.

P5	Any qualifiers valid with the CREATE SERVICE command	LAT$SYSTARTUP.COM uses this argument to assign service characteristics with the LATCP CREATE SERVICE command. You can specify the /IDENTIFICATION, /LOG, and /STATIC_RATING qualifiers. Specify several qualifiers as shown in the following example:
		`"/IDENTIFICATION=" -`
		`  """""Official system node""""" -`
		`  /STATIC_RATING=250"`

Note that if you want to do any of the following LAT network tasks, you must edit LAT $SYSTARTUP.COM (described in *Section 12.6, "Customizing LAT Characteristics"*):

- Set up LAT printers.
- Create special application services.
- Set up the node to allow outgoing connections (to support the SET HOST/LAT command).

For a full description of LATCP commands and qualifiers, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

Example

The following command creates the service OFFICE on the service node MOE, which is part of the OFFICE cluster (refer to *Figure 12.1, "Components of a LAT Network"*):

```
$ @SYS$STARTUP:LAT$STARTUP OFFICE
```

12.6. Customizing LAT Characteristics

To define special LAT characteristics for your node, edit the site-specific command procedure SYS $MANAGER:LAT$SYSTARTUP.COM. This command procedure contains LATCP commands that

define LAT characteristics. LAT$SYSTARTUP.COM is invoked when you execute the LAT$STARTUP command procedure. As explained in *Section 12.5, "Starting Up the LAT Protocol"*, you typically execute LAT$STARTUP.COM from the general-purpose, site-specific startup command procedure.

If you want your node to be a LAT service node that only supports incoming connections from interactive terminals, editing LAT$SYSTARTUP.COM is not necessary. You can assign a service name and other characteristics by specifying parameters when you invoke the command procedure SYS $STARTUP:LAT$STARTUP, as described in *Section 12.5, "Starting Up the LAT Protocol"*.

However, you can edit LAT$SYSTARTUP.COM to add LATCP commands that customize LAT characteristics for your node, for example:

Task	**For More Information**
Create more than one service	*Section 12.6.1, "Creating Additional Services"*
Create logical ports for special application services and printers	*Section 12.6.2, "Setting Up Ports"*
Enable queued incoming requests	*Section 12.6.3, "Queuing Incoming Requests "*
Enable outgoing LAT connections to support the SET HOST/LAT command	*Section 12.6.4, "Enabling Outgoing LAT Connections"*
Tailor node characteristics[1]	*Section 12.6.5, "Sample Edited LAT $SYSTARTUP.COM Procedure"*

[1]For example, to assign special service announcements or LAN links (using the SET NODE and SET LINK commands).

Caution

Do not edit the command procedures LAT$STARTUP.COM and LAT$CONFIG.COM. These are procedures supplied by VSI to perform functions necessary for the LAT protocol to run correctly. Edit only LAT$SYSTARTUP.COM to define LAT characteristics specific to your site.

If you edit LAT$SYSTARTUP.COM, you should add only LATCP commands. In addition, you should conform to the order of commands in the template file SYS$MANAGER: LAT $SYSTARTUP.TEMPLATE. *Section 12.6.5, "Sample Edited LAT$SYSTARTUP.COM Procedure"* provides a sample edited LAT$SYSTARTUP procedure. The *VSI OpenVMS System Management Utilities Reference Manual* contains full descriptions of all the LATCP commands you can include in LAT $SYSTARTUP.COM.

12.6.1. Creating Additional Services

The LAT$SYSTARTUP.COM procedure provided by VSI creates one service. This can be a primary service, one through which users can access the general computing environment. It can also be a special application service, such as a data entry program or an online news service.

You can also create a limited service with a fixed number of LTA devices, as described in *Section 12.6.2.3, "Setting Up Limited Services "*.

The LAT$SYSTARTUP.COM procedure creates the service with the same name as that of your node, unless you specify a unique service name as an argument to the @SYS$STARTUP:LAT $STARTUP.COM command, as explained in *Section 12.5, "Starting Up the LAT Protocol"*.

How to Perform This Task

To create services in addition to the one provided in LAT$SYSTARTUP.COM, use the CREATE SERVICE commands, which you can add to LAT$SYSTARTUP.COM. Note that if you create an application service, VSI recommends that you assign the name of the application program. For more information about the LATCP command CREATE SERVICE, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

Example

The following example creates the application service NEWS on the local node:

```
$ LCP :== $LATCP
$ LCP CREATE SERVICE /APPLICATION NEWS
```

12.6.2. Setting Up Ports

The LAT$SYSTARTUP.COM file provided by VSI includes sample commands to create logical ports on the service node and associates them with physical ports or services on the terminal server node. These ports can be used for application services and remote printers.

How to Perform This Task

To create ports, enable the sample commands by removing the exclamation points (!) that precede them in the LAT$SYSTARTUP.COM file, or add similar CREATE PORT and SET PORT commands to that file to meet your needs. For information about the LATCP commands CREATE PORT and SET PORT, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

Note

VSI strongly recommends that you create application ports and dedicated ports *after* the LATCP command SET NODE/STATE=ON is executed. This minimizes nonpaged pool memory usage and eliminates the possibility of creating duplicate ports.

Note that you may encounter the following error when attempting to create an application port (with a command such as LCP CREATE PORT LTA5001:/APPLICATION, for example):

```
%LAT-W-CMDERROR, error reported by command executor -SYSTEM-F-DUPLNAM,
 duplicate name
```

This error indicates that the LAT application port you are trying to create is already created by some other application. This application could be LATCP itself (LATCP's port, LATCP$MGMT_PORT, is used to communicate with LTDRIVER).

To avoid this error, make sure the SET NODE/STATE=ON command is executed before any commands that create application ports or dedicated ports. You can also use the LATCP command SET NODE/DEVICE_SEED. For more information about the SET NODE/DEVICE_SEED command, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

12.6.2.1. Setting Up Printers

If you set up a port for a printer, you must also perform the following tasks:

1. Create a spooled output queue for the printer.
2. Add a command to start the queue to the startup command procedure that starts your queues, or to the general-purpose, site-specific startup command procedure.

These tasks are described in *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

12.6.2.2. Setting Up Special Application Services

To establish a special application service, include the /DEDICATED qualifier when defining a LAT port. The application program to which the service connects must define the same dedicated port. For example, the following commands set up ports for an application service called NEWS:

```
$ LCP :== $LATCP
$ LCP CREATE PORT LTA333:/DEDICATED
$ LCP SET PORT LTA333:/SERVICE=NEWS
```

Before application services can be available to user terminals on the LAT network, you must start the application program. You usually add commands to SYLOGIN.COM to do this.

12.6.2.3. Setting Up Limited Services

Application services with dedicated ports allow you to create a predetermined number of LTA devices (LAT terminals, for example) that are under the control of a process supplied by the system. In that environment, however, the user cannot log in to the service because no way exists for dedicated LTA devices to run the system login image (LOGINOUT.EXE).

You can create a *limited* service that allows users to log in to a predetermined number of LTA devices associated with that limited service. When all those devices are in use, the LAT software will reject additional connection requests to that service, as indicated by “service in use” error messages. Creating a limited service in this way allows you to control the number of LAT users on your system. (Note, however, that you cannot control which LTA device will be assigned when a user connects to the limited service.)

The following example sets up a limited service with two predetermined LTA devices:

```
$ LCP :== $LATCP
$ LCP CREATE SERVICE /LIMITED RESTRICTED
$ LCP CREATE PORT LTA100 /LIMITED
$ LCP CREATE PORT LTA101 /LIMITED
$ LCP SET PORT LTA100 /SERVICE=RESTRICTED
$ LCP SET PORT LTA101 /SERVICE=RESTRICTED
```

When a user attempts to connect to the limited service named RESTRICTED, the LAT software will choose either LTA100 or LTA101 (whichever is available first) and complete the user connection. The user can then log in to that system. If another user connects to the service, that second connection attempt will be assigned to the remaining LTA device. The user can then log in to that second system. When the two devices associated with the limited service named RESTRICTED are both in use, any subsequent attempts to connect to that service will be rejected, as indicated by the “service in use” error message.

When a user logs out of the system (LTA100 or LTA101), that LTA device is *not* deleted. Instead, it is reset to accept the next connection request to the limited service.

12.6.3. Queuing Incoming Requests

By default, incoming requests to limited or application services are queued. This means that if you attempt to connect to a limited or application service (by using a terminal server port with forward queuing enabled or by entering the DCL command SET HOST/LAT/QUEUE), the LAT software will queue, rather than reject, this connection request if the service has no available ports.

How to Perform This Task

You can set up and manage a service that queues incoming connect requests as follows:

- Use the LATCP command SHOW SERVICE to determine whether the service has queuing enabled or disabled.
- If queuing is disabled, use the SET SERVICE /QUEUED command to enable queuing.
- Use the SET NODE /QUEUE_LIMIT= *n* command on the local node to control the number of free queue slots (where `n` is between 0 and 200).
- Use the SHOW QUEUE_ENTRY [queue-entry-id] command to view entries in the local queue.
- Use the DELETE QUEUE_ENTRY [queue-entry-id] command to delete an entry from the local queue.

Refer to the *VSI OpenVMS System Management Utilities Reference Manual* for more detailed descriptions of the LATCP commands and qualifiers you use to support queued requests.

Example

The following example shows how to enable queuing on your system:

```
$ LCP :== $LATCP
$ LCP SET SERVICE /QUEUE
```

Note

If a system is configured to handle queued connect requests, that system *must* be set up as follows to avoid possible queue connection failures:

- Incoming and outgoing connections must be enabled.
- User group codes and service group codes must be identical.

12.6.4. Enabling Outgoing LAT Connections

By default, outgoing LAT connections are disabled on a node. To allow users to use the SET HOST/LAT connection to establish LAT connections from the node, you must edit LAT$SYSTARTUP.COM to enable outgoing connections. For more details on using the SET HOST/LAT command for outgoing LAT connections, refer to the description of that command in the *VSI OpenVMS DCL Dictionary*.

Commands to enable outgoing connections are included in the LAT$SYSTARTUP.COM file provided by VSI. Enable the command of your choice by removing the exclamation point (!) that precedes it, or add a similar command to meet your needs. For more information, refer to the /CONNECTIONS and /USER_GROUPS qualifiers to the LATCP command SET NODE in the *VSI OpenVMS System Management Utilities Reference Manual*.

To attain optimal SET HOST/LAT performance and forward port performance, you should set the system parameter TTY_ALTYPAHD to 1500 and reboot.

If you want to set up your node only as a service node with incoming connections enabled, editing LAT $SYSTARTUP.COM is not necessary. However, you might edit LAT$SYSTARTUP.COM to do one or more of the following tasks:

- Create more than one service on a node
- Create special application services
- Set up LAT printers
- Enable outgoing LAT connections (to allow a node to act as a terminal server node)
- Tailor node characteristics; for example, to assign special service announcements or connections to the LAN

12.6.5. Sample Edited LAT$SYSTARTUP.COM Procedure

The following example is a sample of an edited LAT$SYSTARTUP.COM procedure that creates services, creates and sets ports, and sets nodes to allow incoming and outgoing connections.

Example 12.1. Sample Edited LAT$SYSTARTUP.COM Procedure

```
$!
$!    LAT$SYSTARTUP.COM - LAT Startup Commands Specific to Site
$!
$!    Use this command procedure to customize the LAT characteristics for
$!    the local node.  These commands, which should serve as examples,
$!    will set up a LAT service name SYS$NODE and default identification
$!    SYS$ANNOUNCE.  The LAT service name and identification will default
$!    to SYS$NODE and SYS$ANNOUNCE unless you specify a service name and
$!    identification as arguments to the command line that invokes
$!    LAT$STARTUP.COM:
$!                    $ @SYS$STARTUP:LAT$STARTUP
$!
$!    You can specify other node and service characteristics (such as
$!    group codes) as arguments to this command line as shown below.
$!
$!        Argument     Function
$!        ---------    --------
$!
$!           P1        Name of the service to be created. If not supplied,
$!                     a service will be created with the same name as
$!                     the node.
$!
$!        P2,P3,P4     Parameters and qualifiers to the SET NODE command.
$!
$!           P5        Parameters and qualifiers to the SET SERVICE
$!                     command. P5 is only used if P1 is specified. More
$!                     than one argument may be supplied by enclosing the
$!                     string in quotes.
$!
$!    Example: $ @SYS$STARTUP:LAT$STARTUP HAWK "/IDENTIFICATION=" -
$!   """""Development node"""""
```

```
$!
$!    Please review and edit this file for possible additions and
$!    deletions that you wish to make.  Future software updates will not
$!    not overwrite the changes made to this file.
$!
$ required_privileges = "OPER"
$ prev_privs = f$setprv(required_privileges)
$ if .not. f$privilege(required_privileges) then goto no_privileges
$ lcp := $latcp
$!
$! -----------------  Modify Node Characteristics  --------------------
$!
$ lcp set node 'p2' 'p3' 'p4'
$!
$! Some examples:
$!
$! ** Allow incoming connections only
$!
$! lcp set node /connections=incoming /groups=(enable=(12,40,43,73),-
_$ disable=0)
$! lcp set node /connections=incoming /groups=enable=(0-255)
$!
$ LCP SET NODE /CONNECTIONS=INCOMING /GROUPS=(ENABLE=(12,40,43,73),-
_$ DISABLE=0)
$!
$! ** Allow outgoing connections only
$!
$! lcp set node /connections=outgoing /user_groups=enable=(24,121-127)
$! lcp set node /connections=outgoing /user_groups=(enable=0-255) -
_$  /node_limit=50
$!
$! ** Enable incoming and outgoing connections
$!
$! lcp set node /connections=both /group=enable=(43,73) /user=enable-
_$ =(44,56)
$! lcp set node /connections=both /group=enable=(0-255) /user=enable-
_$ =(0-255)
$!
$!
$! ---------------  Modify Service Characteristics  -----------------
$!
$ if p1 .eqs. ""
$ then $    lcp create service
$ else
$     lcp create service 'p1' 'p5'
$ endif
$! --------------------  Start LAT Protocol  ------------------------
$!
$ lcp set node /state=on
$!
$!
$! --------------------  Create and Map Ports  ----------------------
$!
$! Some examples:
$!
$! lcp create port lta101: /dedicated
$! lcp create port lta102: /application
$! lcp create port lta103: /application
```

```
$! lcp create port /nolog/logical=(name=ln03$mgmt, table=system,-
_$ mode=executive)
$
$ LCP CREATE PORT LTA1: /NOLOG
$ LCP CREATE PORT LTA20: /NOLOG
$
$! lcp set port lta101: /dedicated /service=graphics
$! lcp set port lta102: /node=server_1 /port=port_1
$! lcp set port lta103: /node=server_2 /service=laser
$! lcp set port ln03$mgmt: /node=server_3 /service=ln03_printers
$! $ LCP SET PORT LTA1: /APPLICATION/NODE=TERM_SERVER_1 /PORT=PORT_6
$ LCP SET PORT LTA20: /APPLICATION/NODE=TERM_SERVER_2 /PORT=PORT_6
$! $exit:
$ prev_privs = f$setprv(prev_privs)
$ exit
$! $no_privileges:
$ write sys$output "Insufficient privileges to execute LATCP commands."
$ write sys$output "Requires ",required_privileges," privileges."
$ goto exit
```

12.7. Managing the LATACP Database Size

On OpenVMS nodes, another component of the LAT software, the LAT Ancillary Control Process (LATACP), maintains the database of available nodes and services. The nodes and services can be those that are multicast from remote LAT nodes, or they can consist of the local node and one or more local services that you create on your own system. The maximum size of this database is dependent on the value of the system parameter CTLPAGES.

After you enter a LATCP command, you might get the following response:

```
%LAT-W-CMDERROR, error reported by command executor
-LAT-F-ACPNOCTL, insufficient resources - ACP CTL/P1 space limit reached
```

If so, this signifies that the database size has reached the CTLPAGES limit. You can correct the situation in one of three ways:

- Reduce the size of the database by reducing the node limit. Use the LATCP command SHOW NODE to display the node limit; use the command SET NODE/NODE_LIMIT to change it. For more information, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

- Reduce the size of the database by reducing the user group codes that are enabled on the node. Use the LATCP command SHOW NODE to display the enabled user group codes; use the command SET NODE/USER_GROUPS=DISABLE to disable some of them. For more information, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

  If you choose this step, you must also edit your startup procedures to change the user groups that are enabled each time the system reboots. For more information, see *Section 12.6, "Customizing LAT Characteristics"*.

- Extend the size of the database by increasing the value of the system parameter CTLPAGES. As a general rule, note that every unit of CTLPAGES that you increase is roughly equivalent to six additional nodes or services that will be stored in the database.

  After you change CTLPAGES, you must reboot the system for the changed value to take effect. Make sure you add the increased value of CTLPAGES to the AUTOGEN parameter file

MODPARAMS.DAT. For more information about changing values of system parameters, see *Section 1.2, "Recommended Method for Changing Parameter Values"*.

Chapter 13. Managing DECdtm Services

This chapter describes what you must do if you want to run software that uses DECdtm services, such as ACMS, DECintact, Oracle Rdb, and RMS Journaling.

Note

On OpenVMS Alpha systems, unpredictable results can occur if DECdtm services are used in a multithreaded environment. Do not make calls to DECdtm services in kernel threads other than the initial thread because much of the work performed by DECdtm uses the context of the calling process.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Planning transaction logs	*Section 13.2, "Planning Transaction Logs"*
Planning for a DECnet-Plus network	*Section 13.3, "Planning for a DECnet-Plus Network"*
Creating transaction logs	*Section 13.4, "Creating Transaction Logs"*
Monitoring transaction performance	*Section 13.5, "Monitoring Transaction Performance"*
Checking whether a transaction log is too small	*Section 13.6, "Checking Whether a Transaction Log Is Too Small"*
Changing the size of a transaction log	*Section 13.7, "Changing the Size of a Transaction Log"*
Moving a transaction log	*Section 13.8, "Moving a Transaction Log"*
Dismounting a disk	*Section 13.9, "Dismounting a Disk"*

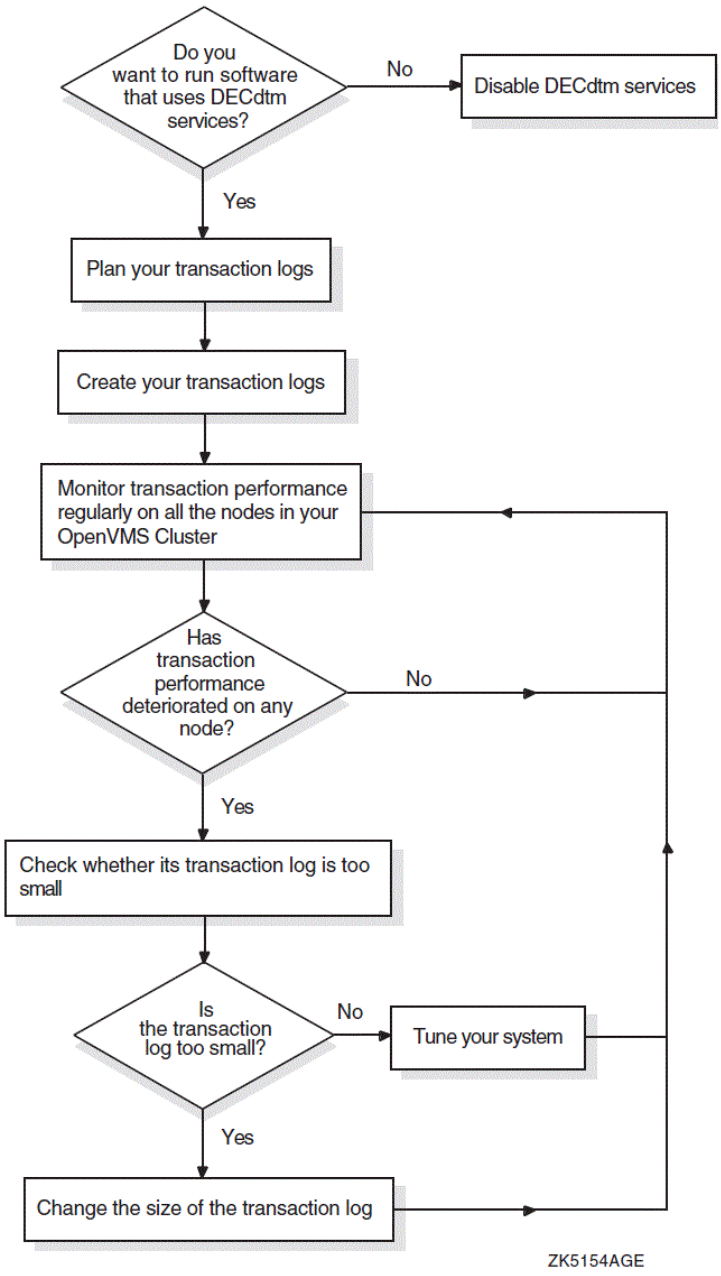
Task	Section
Adding a node	*Section 13.10, "Adding a Node"*
Removing a node	*Section 13.11, "Removing a Node"*
Disabling DECdtm services	*Section 13.12, "Disabling DECdtm Services"*
Enabling DECdtm services	*Section 13.1, "Understanding Transaction Logs"*

The map in *Figure 13.1, "Managing DECdtm Services"* shows the tasks, and the order in which to do them.

This chapter explains the following concepts:

Concept	Section
Understanding transaction logs	*Section 13.1, "Understanding Transaction Logs"*
Understanding transaction groups	*Section 13.3.2.2, "Understanding Transaction Groups"*

Figure 13.1. Managing DECdtm Services



13.1. Understanding Transaction Logs

A **transaction log** is a file that stores information about DECdtm transactions performed on a node. It is of file type .LM$JOURNAL.

Before a node can execute DECdtm transactions, you must create a transaction log for the node. In an OpenVMS Cluster, create a transaction log for each node in the cluster. Use the Log Manager Control Program (LMCP) utility to create and manage transaction logs.

DECdtm services use the logical name SYS$JOURNAL to find transaction logs. You must define SYS $JOURNAL to point to the directories that contain transaction logs.

13.2. Planning Transaction Logs

The size and location of a transaction log can affect transaction performance. Before you create a transaction log, decide the size and location of the transaction log.

Later, you can change the size of a transaction log, or move it. However, careful planning at this stage reduces the need for future changes.

This section describes:

Task	Section
Deciding the size of a transaction log	*Section 13.2.1, "Deciding the Size of a Transaction Log"*
Deciding the location of a transaction log	*Section 13.2.2, "Deciding the Location of a Transaction Log"*

13.2.1. Deciding the Size of a Transaction Log

When you create a transaction log, you can specify its size. The default size is 4000 blocks; this gives acceptable performance on most systems.

If you know the expected rate of transactions, VSI suggests the following formula to calculate the transaction log size:

```
size = 40 * rate
```

where:

size	is the size of the transaction log in blocks.
rate	is the average number of transactions executed per second.

If you do not know the rate of transactions, accept the default size of 4000 blocks.

13.2.2. Deciding the Location of a Transaction Log

If possible, choose a disk that is:

Fast	Achieve speed by using a high-performance disk, such as a solid-state disk, that is not heavily used.
Highly available	Achieve high availability by having multiple access paths to the data. In an OpenVMS Cluster, use a disk that can be accessed by the other nodes in the cluster. This ensures that if one node fails, transactions running on other nodes are not blocked.
Reliable	Achieve reliability by keeping multiple copies of the data. Using a shadowed disk is more reliable than using a nonshadowed disk, but may be slower because transaction logs are almost exclusively write-only.

You may need to choose between speed and either availability or reliability. For example, if the node is a workstation, you may choose to sacrifice speed for availability and reliability by putting the node's transaction log on a shadowed HSC-based disk, instead of on a faster disk attached to the workstation.

In a cluster environment, try to distribute the transaction logs across different disks. Having more than one transaction log on a disk can lead to poor transaction performance.

Note

Make sure that the disk has enough contiguous space to hold the transaction log. A discontiguous transaction log leads to poor transaction performance.

13.3. Planning for a DECnet-Plus Network

This section contains the following information to help you plan for using DECdtm in a DECnet-Plus network:

- Planning your DECnet-Plus namespace
- Planning SCSNODE names in your DECnet-Plus network

13.3.1. Planning Your DECnet-Plus Namespace

DECdtm does not support multiple DECnet-Plus namespaces.

This means that if you want to use software that uses DECdtm services, you cannot use both a local namespace and a DECdns namespace.

13.3.2. Planning SCSNODE Names in Your DECnet-Plus Network

SCSNODE is a system parameter that defines the name of the computer. You must follow certain rules when choosing SCSNODE names if you have a DECnet-Plus network and you want to perform DECdtm transactions that span either different OpenVMS Clusters or different standalone computers.

13.3.2.1. Rules for SCSNODE Names

If you have a DECnet-Plus network and want to perform DECdtm transactions that span different OpenVMS Clusters or different standalone computers, you must make sure that your SCSNODE names obey the following rules:

- The SCSNODE name of each computer in a transaction group must be different from:
  - The SCSNODE names of the other computers in the transaction group; SCSNODE names must be unique within a transaction group
  - The DECnet simple names of other computers on the same local root
  - The DECnet synonyms of the other computers in the entire network

  For an explanation of transaction groups, see *Section 13.3.2.2, "Understanding Transaction Groups"*.
- If a computer is in an OpenVMS Cluster, its SCSNODE name must also be different from:

- The DECnet simple names of the other computers in the same cluster
- The DECnet simple names of computers on the same local root as other cluster members

13.3.2.2. Understanding Transaction Groups

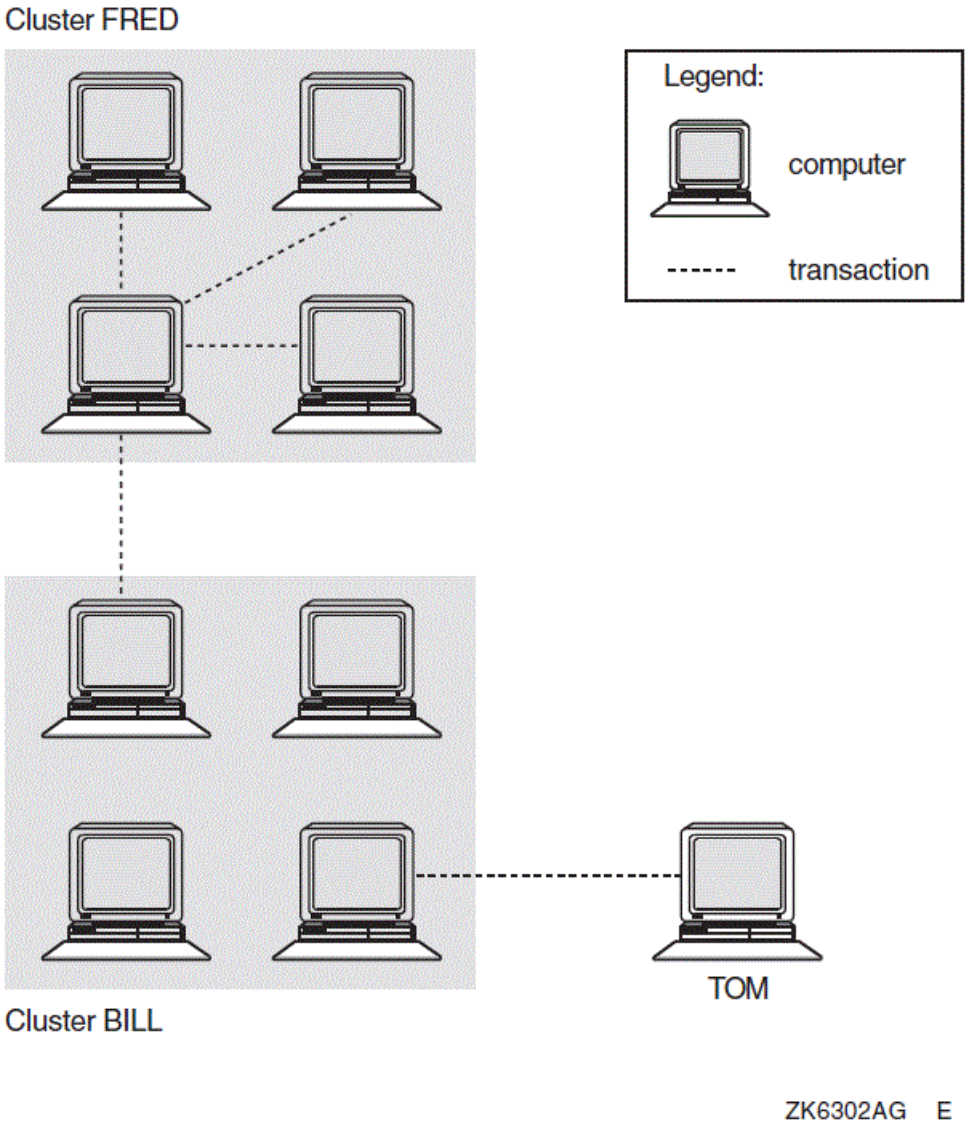
A **transaction group** is a group of computers involved in DECdtm transactions whose SCSNODE names must obey the rules described in *Section 13.3.2.1, "Rules for SCSNODE Names"*.

A transaction group conforms to the following guidelines:

- Each computer belongs to no more than one transaction group.
- All the computers in an OpenVMS Cluster belong to the same transaction group.
- If a single transaction spans computers A and B, then computers A and B belong to the same transaction group.

Figure 13.2, "Transaction Group" shows an example of a transaction group.

Figure 13.2. Transaction Group



All nine computers shown in the figure are in the same transaction group because:

- A transaction spans a computer in cluster FRED and a computer in cluster BILL. This means that the four computers in cluster FRED and the four computers in cluster BILL are in the same transaction group.
- A transaction spans standalone computer TOM and a computer in cluster BILL. This means that the standalone computer TOM is in the same transaction group as the computers in cluster BILL.

13.4. Creating Transaction Logs

Before a node can perform DECdtm transactions, you must create a transaction log for the node. In an OpenVMS Cluster environment, create a transaction log for each node.

Caution

Removing a node from a cluster after you have created the transaction logs can lead to data corruption. For instructions on how to remove a node safely, see *Section 13.11, "Removing a Node"*.

How to Perform This Task

1. For each node, decide the size and location of the transaction log, using the guidelines in *Section 13.2, "Planning Transaction Logs"*. Remember that the disks must have enough contiguous space to hold the transaction logs.
2. If you are in a cluster environment, make sure that the disks on which you want to create the transaction logs are mounted clusterwide.
3. Decide in which directories you want to create the transaction logs. You may want to create new directories for the transaction logs.
4. Define SYS$JOURNAL to point to the directories in which you want to create the transaction logs:

   ```
   DEFINE/SYSTEM/EXECUTIVE_MODE SYS$JOURNAL dirspec[,...]
   ```

   where *dirspec* is the full specification of a directory in which you want to create one or more transaction logs. List all the directories that will contain transaction logs. You can list the directories in any order.

   In a cluster environment, use SYSMAN to define SYS$JOURNAL clusterwide.
5. Edit the SYS$MANAGER:SYLOGICALS.COM command procedure to include the SYS $JOURNAL definition.

   If you created node-specific versions of SYLOGICALS.COM, edit all the versions.
6. Create one transaction log for each node, using LMCP's CREATE LOG command:

   ```
   CREATE LOG [/SIZE=size] dirspecSYSTEM$node.LM$JOURNAL
   ```

   where:

size	is the size of the transaction log in blocks. By default, the size of the transaction log is 4000 blocks.

dirspec	is the full specification of the directory in which you want to create the transaction log.
node	is the name of the node.

7. Make sure DECdtm services are enabled as follows:

Step	Action	
a.	Check whether the logical SYS$DECDTM_INHIBIT is defined: `$ SHOW LOGICAL SYS$DECDTM_INHIBIT`	
b.	Is SYS$DECDTM_INHIBIT defined?	
	Yes	DECdtm services are disabled. Enable DECdtm services by following the instructions in *Section 13.1, "Understanding Transaction Logs"*.
	No	DECdtm services are enabled.

Example

This example shows how to create transaction logs in an OpenVMS Cluster that consists of two nodes whose SCSNODE names are BLUE and RED. Neither node has a node-specific version of SYLOGICALS.COM.

Decide the size and location of the transaction logs:

Node	Size of Log (in Blocks)	Disk
BLUE	5000	DUA1
RED	4000	DUA2

Mount the disks clusterwide:

```
$ MOUNT/CLUSTER/SYSTEM DUA1: LOG1
$ MOUNT/CLUSTER/SYSTEM DUA2: LOG2
```

Create directories for the transaction logs:

```
$ CREATE/DIRECTORY DISK$LOG1:[LOGFILES]
$ CREATE/DIRECTORY DISK$LOG2:[LOGFILES]
```

Define SYS$JOURNAL:

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> SET ENVIRONMENT/CLUSTER
SYSMAN> DO DEFINE/SYSTEM/EXECUTIVE_MODE SYS$JOURNAL -
_SYSMAN> DISK$LOG1:[LOGFILES], DISK$LOG2:[LOGFILES]
SYSMAN> EXIT
```

Edit the SYS$MANAGER:SYLOGICALS.COM command procedure to include the following line:

```
$ !
$ DEFINE/SYSTEM/EXECUTIVE_MODE SYS$JOURNAL DISK$LOG1:[LOGFILES], -
DISK$LOG2:[LOGFILES]
$ !
```

Create the transaction logs:

```
$ RUN SYS$SYSTEM:LMCP
```

```
LMCP> CREATE LOG/SIZE=5000 DISK$LOG1:[LOGFILES]SYSTEM$BLUE.LM$JOURNAL
LMCP> CREATE LOG DISK$LOG2:[LOGFILES]SYSTEM$RED.LM$JOURNAL
LMCP> EXIT
```

Make sure DECdtm services are enabled:

```
$ SHOW LOGICAL SYS$DECDTM_INHIBIT
%SHOW-S-NOTRAN, no translation for logical name SYS$DECDTM_INHIBIT
```

SYS$DECDTM_INHIBIT is undefined, so DECdtm services are enabled.

13.5. Monitoring Transaction Performance

Changes to your system, such as increase in work load, can affect transaction performance. Once a month, monitor transactions on the node to make sure that transaction performance has not deteriorated. In an OpenVMS Cluster environment, monitor transaction performance on all the nodes in the cluster.

How to Perform This Task

1. Monitor transactions, using the MONITOR TRANSACTION command:

   ```
   MONITOR TRANSACTION/SUMMARY[=file-spec]/ENDING=time/NODE=(nodename,...)
   ```

   where:

file-spec	is the file specification of the summary file. Information about transactions is summarized and recorded in the summary file. If you omit the file specification, the information is recorded in MONITOR.SUM in your default directory.
time	is the time that the monitoring session ends.
nodename	is the name of a node. In an OpenVMS Cluster, list all the nodes in the cluster.

   For the best results, monitor transactions for a day at a time.

   You can monitor transactions in batch mode by including the MONITOR TRANSACTION command in a command procedure.

   For a full description of the MONITOR TRANSACTION command, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

2. Examine the summary file.

   The summary file contains values for a number of different data items. Note the following values for each node:

   - Average end rate. This is the average number of transactions completed per second.
   - Average completion rates. These are the average numbers of transactions completed in the following times:

     Less than 1 second
     Between 1 and 2 seconds
     Between 2 and 3 seconds
     Between 3 and 4 seconds

Between 4 and 5 seconds
More than 5 seconds

Keep a note of these values.

3. Compare the results from this monitoring session with the results from previous sessions.

   For the same work load, the rate and duration of transactions should remain about the same. Indications of performance deterioration are:

   - The average end rate decreases
   - The average duration increases

     To find out whether the average duration of transactions has increased, compare the average completion rates. If a greater proportion of the transactions takes longer to complete, the average duration of transactions has increased.

   Note any trends over a number of monitoring sessions. Variations from one monitoring session to the next are probably due to variations in work load.

   If you suspect that transaction performance has deteriorated on any node, check whether its transaction log is too small (see *Section 13.6, "Checking Whether a Transaction Log Is Too Small"*).

   If the transaction log is big enough, but transaction performance still deteriorates, tuning the system might be necessary. Refer to the *VSI OpenVMS Cluster Systems Manual* for information about tuning your system.

Example

This example shows how to monitor transaction performance on an OpenVMS Cluster that consists of two nodes whose SCSNODE names are BLUE and RED.

Monitor transactions on nodes BLUE and RED for one day:

```
$ MONITOR TRANSACTION/SUMMARY=DISK$LOG1:[LOGFILES]TRANSACTIONS.SUM -
_$ /ENDING="+1-"/NODE=(BLUE,RED)
```

Examine the summary file:

```
                    DISTRIBUTED TRANSACTION STATISTICS
                              on node BLUE         From: 16-OCT-2016
 14:23:51
                                 SUMMARY           To:   17-OCT-2016
 14:23:51
                                    CUR        AVE        MIN        MAX

Start Rate                        49.02      43.21      31.30      49.02
Prepare Rate                      48.70      43.23      30.67      48.70
One Phase Commit Rate              0.00       0.00       0.00       0.00
Total Commit Rate                 48.70      43.19      31.30      48.70
Abort Rate                         0.00       0.00       0.00       0.00
End Rate                          48.70      43.19      31.30      48.70
Remote Start Rate                  0.00       0.00       0.00       0.00
Remote Add Rate                    0.00       0.00       0.00       0.00

Completion Rate    0-1            21.42      13.57       0.63      21.42
```

```
by Duration       1-2          25.97      29.15      24.59      33.87
in Seconds        2-3           1.29       0.47       0.00       4.47
                  3-4           0.00       0.00       0.00       0.00
                  4-5           0.00       0.00       0.00       0.00
                   5+           0.00       0.00       0.00       0.00

                                SUMMARIZING


                      DISTRIBUTED TRANSACTION STATISTICS
                                on node RED          From: 16-OCT-2016
14:23:52
                                   SUMMARY           To:   17-OCT-2016
14:23:52
   .
   .
   .
```

Make a note of the following values:

- Average end rate.

  For node BLUE, the average end rate is 43.19 transactions per second.

- Average completion rates.

  For node BLUE, the average completion rates are as follows:

  13.57 transactions completed in 0 to 1 seconds
  29.15 transactions completed in 1 to 2 seconds
  0.47 transactions completed in 2 to 3 seconds

Compare the results from this monitoring session to those of previous sessions:

Session	**End Rate**	**Completion Rates**		
		0--1 Secs	**1--2 Secs**	**2--3 Secs**
June	42.13	12.98	28.13	1.02
July	38.16	10.35	25.80	2.01
August	43.19	13.57	29.15	0.47

The results for node BLUE show no signs of deteriorating performance.

13.6. Checking Whether a Transaction Log Is Too Small

If transaction performance has deteriorated on a node, check whether its transaction log is too small.

Section 13.5, "Monitoring Transaction Performance" describes how to find out whether transaction performance has deteriorated.

How to Perform This Task

1. Log in to the node that the transaction log belongs to.

2. Check how many times the transaction log has stalled, using LMCP's SHOW LOG/CURRENT command:

   ```
   $ RUN SYS$SYSTEM:LMCP
   LMCP> SHOW LOG/CURRENT
   ```

   Note the number of checkpoints and stalls displayed by this command.

3. Wait for five minutes, then repeat the SHOW LOG/CURRENT command. Note the number of checkpoints and stalls again.

4. Compare the information from the SHOW LOG/CURRENT commands:

   If the number of checkpoints has not changed, wait until the system is busier, then try this task again.

   If the number of checkpoints has increased, and the number of stalls has increased by more than one, the transaction log is too small.

5. If the transaction log is too small, increase its size. For information about how to change the size of a transaction log, see *Section 13.7, "Changing the Size of a Transaction Log"*.

Example

This example shows how to check whether node BLUE's transaction log is too small.

Log in to node BLUE. Then check how many times the transaction log has stalled:

```
$ RUN SYS$SYSTEM:LMCP
LMCP> SHOW LOG/CURRENT
Checkpoint starts/ends            2464/2464
Stall starts/ends                   21/21
Log status: no checkpoint in progress, no stall in progress.
```

The number of checkpoints is 2464, and the transaction log has stalled 21 times.

Wait for five minutes, then repeat the SHOW LOG/CURRENT command:

```
LMCP> SHOW LOG/CURRENT
Checkpoint starts/ends            2514/2514
Stall starts/ends                   28/28
Log status: no checkpoint in progress, no stall in progress.
```

The number of checkpoints has increased since the previous reading, and the transaction log has now stalled 28 times, an increase of 7. This means that the transaction log is too small.

13.7. Changing the Size of a Transaction Log

To determine if changing the size of a transaction log is necessary, see *Section 13.6, "Checking Whether a Transaction Log Is Too Small"*.

How to Perform This Task

Caution

Follow all the steps carefully. Taking shortcuts can lead to data corruption.

1. Log in to the node that the transaction log belongs to.
2. Find out which directory the transaction log is in, using LMCP's SHOW LOG command:

   ```
   SHOW LOG SYSTEM$node.LM$JOURNAL
   ```

   where *node* is the name of the node that the transaction log belongs to.
3. Rename the transaction log:

   ```
   RENAME dirspecSYSTEM$node.LM$JOURNAL dirspecSYSTEM$node.LM$OLD
   ```

   where:

dirspec	is the full specification of the directory containing the transaction log.
node	is the name of the node that the transaction log belongs to.

4. Can you stop all the software that uses DECdtm services without shutting down any nodes?

<table>
<tr><td>Yes</td><td colspan="3">Close the transaction log as follows:</td></tr>
<tr><td></td><td>Step</td><td colspan="2">Action</td></tr>
<tr><td></td><td>a.</td><td colspan="2">Stop all the software that uses DECdtm services.</td></tr>
<tr><td></td><td>b.</td><td colspan="2">Close the transaction log using LMCP's CLOSE LOG command:

<code>$ RUN SYS$SYSTEM:LMCP
LMCP> CLOSE LOG</code>

The CLOSE LOG command closes the transaction log and stops the DECdtm TP_SERVER process. The command fails if any software is using DECdtm services.</td></tr>
<tr><td></td><td>c.</td><td colspan="2">Did the CLOSE LOG command succeed?</td></tr>
<tr><td></td><td></td><td>Yes</td><td>Restart the TP_SERVER process:

<code>$ @SYS$STARTUP:DECDTM$STARTUP.COM</code></td></tr>
<tr><td></td><td></td><td>No</td><td>Wait for 30 seconds, then repeat steps 4b and 4c.</td></tr>
<tr><td>No</td><td colspan="3">Close the transaction log by rebooting the node. Log in to the node when it has rebooted.</td></tr>
</table>

5. Change the size of the transaction log, using LMCP's CONVERT LOG command:

   ```
   CONVERT LOG/SIZE=size dirspec SYSTEM$node.LM$OLD dirspec SYSTEM$node.LM
   $JOURNAL
   ```

   where:

size	is the new size of the transaction log in blocks.
dirspec	is the full specification of the directory containing the transaction log.
node	is the name of the node that the transaction log belongs to.

6. If you stopped the software that uses DECdtm services in step 4, restart the software.
7. Delete the old transaction log:

   ```
   DELETE dirspec SYSTEM$node.LM$OLD;
   ```

where:

dirspec	is the full specification of the directory containing the old transaction log.
node	is the name of the node that the transaction log belongs to.

Example

This example shows how to change the size of node RED's transaction log to 6000 blocks. Node RED is in an OpenVMS Cluster, and its transaction log is in DISK$LOG2:[LOGFILES].

Log in to node RED. Find out which directory RED's transaction log is in, then rename the transaction log:

```
$ RUN SYS$SYSTEM:LMCP
LMCP> SHOW LOG SYSTEM$RED.LM$JOURNAL
Directory of DISK$LOG2:[LOGFILES]

SYSTEM$RED.LM$JOURNAL;1

Total of 1 file.
LMCP> EXIT

$ RENAME DISK$LOG2:[LOGFILES]SYSTEM$RED.LM$JOURNAL -
_$ DISK$LOG2:[LOGFILES]SYSTEM$RED.LM$OLD
```

Stop all software that uses DECdtm services. Then close the transaction log:

```
$ RUN SYS$SYSTEM:LMCP
LMCP> CLOSE LOG
Transaction log closed, TP_SERVER process stopped
LMCP> EXIT
```

Restart the TP_SERVER process:

```
$ @ SYS$STARTUP:DECDTM$STARTUP.COM
```

Change the size of the transaction log:

```
$ RUN SYS$SYSTEM:LMCP
LMCP> CONVERT LOG/SIZE=6000 DISK$LOG2:[LOGFILES]SYSTEM$RED.LM$OLD -
_LMCP> DISK$LOG2:[LOGFILES]SYSTEM$RED.LM$JOURNAL
Log file DISK$LOG2:[LOGFILES]SYSTEM$RED.LM$JOURNAL;1 created.
Log file DISK$LOG2:[LOGFILES]SYSTEM$RED.LM$OLD converted.
LMCP> EXIT
```

Restart the software that uses DECdtm services.

Delete the old transaction log:

```
$ DELETE DISK$LOG2:[LOGFILES]SYSTEM$RED.LM$OLD;
```

13.8. Moving a Transaction Log

You may want to move a transaction log if:

- You want to place the transaction log on a faster disk

- You want to redistribute the work load on your disks

How to Perform This Task

Caution

Follow all the steps carefully. Taking shortcuts can lead to data corruption.

1. Decide the location that you want to move the transaction log to, using the guidelines in *Section 13.2.2, "Deciding the Location of a Transaction Log"*. Remember that the disk must have enough contiguous space to hold the transaction log.

2. Log in to the node that the transaction log belongs to.

3. If you are in an OpenVMS Cluster, make sure that the disk you want to move the transaction log to is mounted clusterwide.

4. Decide which directory you want to move the transaction log to. You may want to create a new directory for the transaction log.

5. Find out which directory the transaction log is in, using LMCP's SHOW LOG command:

   ```
   SHOW LOG SYSTEM$node.LM$JOURNAL
   ```

   where *node* is the name of the node that the transaction log belongs to.

6. Rename the transaction log:

   ```
   RENAME dirspec SYSTEM$node.LM$JOURNAL dirspecSYSTEM$node.LM$OLD
   ```

   where:

dirspec	is the full specification of the directory containing the transaction log.
node	is the name of the node that the transaction log belongs to.

7. Can you stop all the software that uses DECdtm services without shutting down any nodes?

<table>
<tr><td rowspan="4">Yes</td><td colspan="3">Close the transaction log as follows:</td></tr>
<tr><td>Step</td><td colspan="2">Action</td></tr>
<tr><td>a.</td><td colspan="2">Stop all the software that uses DECdtm services.</td></tr>
<tr><td>b.</td><td colspan="2">Close the transaction log using LMCP's CLOSE LOG command:

<code>$ RUN SYS$SYSTEM:LMCP</code>

<code>LMCP> CLOSE LOG</code>

The CLOSE LOG command closes the transaction log and stops the DECdtm TP_SERVER process. The command fails if any software is using DECdtm services.</td></tr>
<tr><td rowspan="2">c.</td><td>Yes</td><td>Restart the TP_SERVER process:

<code>$ @SYS$STARTUP:DECDTM$STARTUP.COM</code></td></tr>
<tr><td>No</td><td>Wait for 30 seconds, then repeat steps 7b and 7c.</td></tr>
</table>

No	Close the transaction log by rebooting the node. Log in to the node when it has rebooted.

8. Make sure that SYS$JOURNAL points to the directory that you want to move the log to. If SYS $JOURNAL does not point to this directory, redefine SYS$JOURNAL:

   ```
   DEFINE/SYSTEM/EXECUTIVE_MODE SYS$JOURNAL dirspec[,...]
   ```

   where *dirspec* is the full specification of a directory containing one or more transaction logs. List all the directories that will contain transaction logs after you have moved the transaction log. You can list the directories in any order.

   In an OpenVMS Cluster, use SYSMAN to redefine SYS$JOURNAL clusterwide.

9. If you redefined SYS$JOURNAL in step 8, edit the SYS$MANAGER:SYLOGICALS.COM command procedure to update the definition of SYS$JOURNAL.

   If you created node-specific versions of SYLOGICALS.COM, edit all the versions.

10. Move the transaction log, using LMCP's CONVERT LOG command:

    ```
    CONVERT LOG old-dirspec SYSTEM$node.LM$OLD new-dirspec SYSTEM$node.LM
    $JOURNAL
    ```

    where:

old-dirspec	is the full specification of the directory that currently contains the transaction log.
node	is the name of the node that the transaction log belongs to.
new-dirspec	is the full specification of the directory that you are moving the transaction log to.

11. If you stopped the software that uses DECdtm services in step 7, restart the software.

12. Delete the old transaction log:

    ```
    DELETE dirspec SYSTEM$node.LM$OLD;
    ```

    where:

dirspec	is the full specification of the directory containing the old transaction log.
node	is the name of the node that the transaction log belongs to.

Example

This example shows how to move BLUE's transaction log. BLUE is in an OpenVMS Cluster. The cluster members and the locations of their transaction logs are as follows:

Node	**Directory Containing Log**
BLUE	DISK$LOG1:[LOGFILES]
RED	DISK$LOG2:[LOGFILES]

Neither node has a node--specific version of SYLOGICALS.COM.

Decide where you want to move BLUE's transaction log to. In this example, assume that you want to move it to DISK$LOG3:[LOGFILES].

Log in to node BLUE. Then mount the disk clusterwide, and create a new directory for the transaction log:

```
$ MOUNT/CLUSTER/SYSTEM DUA3: LOG3
$ CREATE/DIRECTORY DISK$LOG3:[LOGFILES]
```

Find out which directory BLUE's transaction log is in, then rename the transaction log:

```
$ RUN SYS$SYSTEM:LMCP
LMCP> SHOW LOG SYSTEM$BLUE.LM$JOURNAL
Directory of DISK$LOG1:[LOGFILES]

SYSTEM$BLUE.LM$JOURNAL;1

Total of 1 file.
LMCP> EXIT
$ RENAME DISK$LOG1:[LOGFILES]SYSTEM$BLUE.LM$JOURNAL -
_$ DISK$LOG1:[LOGFILES]SYSTEM$BLUE.LM$OLD
```

Stop all software that uses DECdtm services. Then close the transaction log:

```
$ RUN SYS$SYSTEM:LMCP
LMCP> CLOSE LOG
Transaction log closed, TP_SERVER process stopped
LMCP> EXIT
```

Restart the TP_SERVER process:

```
$ @SYS$STARTUP:DECDTM$STARTUP.COM
```

Redefine SYS$JOURNAL:

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> SET ENVIRONMENT/CLUSTER
SYSMAN> DO DEFINE/SYSTEM/EXECUTIVE_MODE SYS$JOURNAL -
_SYSMAN> DISK$LOG2:[LOGFILES], DISK$LOG3:[LOGFILES]
SYSMAN> EXIT
```

Edit the SYS$MANAGER:SYLOGICALS.COM command procedure to update the SYS$JOURNAL definition. Then move the transaction log:

```
$ RUN SYS$SYSTEM:LMCP
LMCP> CONVERT LOG DISK$LOG1:[LOGFILES]SYSTEM$BLUE.LM$OLD -
_LMCP> DISK$LOG3:[LOGFILES]SYSTEM$BLUE.LM$JOURNAL
Log file DISK$LOG3:[LOGFILES]SYSTEM$BLUE.LM$JOURNAL;1 created.
Log file DISK$LOG1:[LOGFILES]SYSTEM$BLUE.LM$OLD converted.
LMCP> EXIT
```

Restart the software that uses DECdtm services. Then delete the old transaction log:

```
$ DELETE DISK$LOG1:[LOGFILES]SYSTEM$BLUE.LM$OLD;
```

13.9. Dismounting a Disk

Before you can dismount a disk, you must close any transaction logs on the disk.

This section describes how to dismount a disk that has transaction logs.

How to Perform This Task

1. Find out which transaction logs are on the disk you want to dismount, using LMCP's SHOW LOG command:

   ```
   $ RUN SYS$SYSTEM:LMCP
   LMCP> SHOW LOG
   ```

2. Stop all the software that uses DECdtm services, if you can do so without shutting down any nodes.

   If you cannot stop the software, reboot one or more nodes in step 3.

3. For each transaction log on the disk:

   a. Log in to the node that the transaction log belongs to.

   b. Rename the transaction log:

      ```
      RENAME dirspecSYSTEM$node.LM$JOURNAL dirspecSYSTEM$node.LM$TEMP
      ```

      where:

dirspec	is the full specification of the directory containing the transaction log.
node	is the name of the node that the transaction log belongs to.

   c. Did you stop all the software that uses DECdtm services in step 2?

<table>
<tr><td>Yes</td><td colspan="3">Close the transaction log as follows:</td></tr>
<tr><td></td><td>Step</td><td colspan="2">Action</td></tr>
<tr><td></td><td>1)</td><td colspan="2">Close the transaction log using LMCP's CLOSE LOG command:

<code>$ RUN SYS$SYSTEM:LMCP</code>

<code>LMCP> CLOSE LOG</code>

The CLOSE LOG command closes the transaction log, and stops the DECdtm TP_SERVER process. The command fails if any software is using DECdtm services.</td></tr>
<tr><td></td><td>2)</td><td colspan="2">Did the CLOSE LOG command succeed?</td></tr>
<tr><td></td><td></td><td>Yes</td><td>Restart the TP_SERVER process:

<code>$ @SYS$STARTUP:DECDTM$STARTUP.COM</code></td></tr>
<tr><td></td><td></td><td>No</td><td>Wait for 30 seconds, then repeat step 3c.</td></tr>
<tr><td>No</td><td colspan="3">Close the transaction log by rebooting the node. Log in to the node when it has rebooted.</td></tr>
</table>

4. Dismount the disk. For instructions on how to dismount a disk, see *Section 13.9, "Dismounting a Disk"*.

5. When you want to mount the disk again, follow these steps:

   a. Mount the disk. For instructions on how to mount a disk, see Section 9.5.

If you are in a cluster, mount the disk clusterwide.

b. Rename each transaction log on the disk:

```
RENAME dirspecSYSTEM$node.LM$TEMP dirspecSYSTEM$node.LM$JOURNAL
```

where:

dirspec	is the full specification of the directory containing the transaction log.
node	is the name of the node that the transaction log belongs to.

c. If you stopped the software that uses DECdtm services, restart the software.

Example

This example shows how to dismount the disk DISK$LOG3.

Find out which transaction logs are on the disk:

```
$ RUN SYS$SYSTEM:LMCP
LMCP> SHOW LOG
    .
    .
    .
Directory of DISK$LOG3:[LOGFILES]

SYSTEM$BLUE.LM$JOURNAL;1)
```

The only transaction log on DISK$LOG3 is node BLUE's transaction log.

Stop all the software that uses DECdtm services.

Log in to node BLUE. Then rename the transaction log:

```
$ RENAME DISK$LOG3:[LOGFILES]SYSTEM$BLUE.LM$JOURNAL -
_$ DISK$LOG3:[LOGFILES]SYSTEM$BLUE.LM$TEMP
```

Close the transaction log:

```
$ RUN SYS$SYSTEM:LMCP
LMCP> CLOSE LOG
Transaction log closed, TP_SERVER process stopped
LMCP> EXIT
```

Restart the TP_SERVER process:

```
$ @SYS$STARTUP:DECDTM$STARTUP.COM
```

Dismount the disk:

```
$ DISMOUNT/CLUSTER DISK$LOG3:
```

When you want to mount the disk again, mount it clusterwide:

```
$ MOUNT/CLUSTER/SYSTEM DUA3: LOG3
```

Rename BLUE's transaction log:

```
$ RENAME DISK$LOG3:[LOGFILES]SYSTEM$BLUE.LM$TEMP -
_$ DISK$LOG3:[LOGFILES]SYSTEM$BLUE.LM$JOURNAL
```

Restart the software that uses DECdtm services.

13.10. Adding a Node

For every node you add to an OpenVMS Cluster, you must create a new transaction log. This section describes how to create a transaction log for a new node.

How to Perform This Task

Before you perform this task, the new node must be configured into the cluster. For instructions on how to configure a node into a cluster, refer to *VSI OpenVMS Cluster Systems Manual*.

1. Decide the size and location of the new node's transaction log, using the guidelines in *Section 13.2, "Planning Transaction Logs"*. Remember that the disk must have enough contiguous space to hold the log.

2. Make sure that the disk on which you want to create the transaction log is mounted clusterwide.

3. Decide which directory you want to create the new transaction log in. You may want to create a new directory for the transaction log.

4. Make sure that SYS$JOURNAL points to the directory in which you want to create the new node's transaction log. If SYS$JOURNAL does not point to this directory, use SYSMAN to redefine SYS $JOURNAL clusterwide:

   ```
   DO DEFINE/SYSTEM/EXECUTIVE_MODE SYS$JOURNAL dirspec[,...]
   ```

   where *dirspec* is the full specification of a directory containing one or more transaction logs. List all the directories that contain transaction logs, including the directory in which you want to create the new node's transaction log. You can list the directories in any order.

5. If you redefined SYS$JOURNAL in step 4, edit the SYS$MANAGER:SYLOGICALS.COM command procedure to update the SYS$JOURNAL definition.

   If you created node-specific versions of SYLOGICALS.COM, edit all the versions.

6. Create the transaction log, using LMCP's CREATE LOG command:

   ```
   CREATE LOG [/SIZE=size] dirspecSYSTEM$node.LM$JOURNAL
   ```

   where:

size	is the size of the transaction log in blocks. By default, the size of the transaction log is 4000 blocks.
dirspec	is the full specification of the directory in which you want to create the transaction log.
node	is the name of the new node.

Example

This example shows how to create a transaction log for a new node, whose SCSNODE name is WHITE.

In this example, the cluster members and the locations of their transaction logs are as follows:

Node	Directory Containing Log
BLUE	DISK$LOG3:[LOGFILES]
RED	DISK$LOG2:[LOGFILES]

Neither node has a node-specific version of SYLOGICALS.COM.

Decide the size and location of WHITE's transaction log:

Node	Size of Log (in Blocks)	Disk
WHITE	5000	DUA4

Mount the disk DUA4 clusterwide:

```
$ MOUNT/CLUSTER/SYSTEM DUA4: LOG4
```

Create a directory for the transaction log:

```
$ CREATE/DIRECTORY DISK$LOG4:[LOGFILES]
```

Redefine SYS$JOURNAL:

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> SET ENVIRONMENT/CLUSTER
SYSMAN> DO DEFINE/SYSTEM/EXECUTIVE_MODE SYS$JOURNAL -
_SYSMAN> DISK$LOG2:[LOGFILES], DISK$LOG3[LOGFILES], DISK$LOG4:[LOGFILES]
SYSMAN> EXIT
```

Edit the SYS$STARTUP:SYLOGICALS command procedure to update the SYS$JOURNAL definition. Then create the transaction log:

```
$ RUN SYS$SYSTEM:LMCP
LMCP> CREATE LOG/SIZE=5000 DISK$LOG4:[LOGFILES]SYSTEM$WHITE.LM$JOURNAL
LMCP> EXIT
```

13.11. Removing a Node

This section describes how to remove a node if you are using DECdtm services.

How to Perform This Task

If you have a standalone machine, perform steps 1 to 8 only.

Caution

Follow all the steps carefully. Taking shortcuts can lead to data corruption.

1. Log in to the node that you want to remove.
2. Stop all the software that uses DECdtm services.
3. Find out whether the node's transaction log contains any active transactions, using LMCP's DUMP/ACTIVE command:

```
DUMP/ACTIVE SYSTEM$node.LM$JOURNAL
```

where *node* is the name of the node that you want to remove.

This command displays details of all the active transactions. The last line gives the total number of active transactions.

4. If the transaction log contains active transactions, follow these steps:
   a. Run recovery procedures for all software that uses DECdtm services.
   b. Find out if the node's transaction log still contains active transactions, using LMCP's DUMP/ACTIVE command.
   c. If the transaction log still contains active transactions, contact your VSI support representative.
5. Redefine SYS$JOURNAL to exclude the directory that contains the transaction log of the node you want to remove, unless the directory contains other transaction logs.

   ```
   DEFINE/SYSTEM/EXECUTIVE_MODE SYS$JOURNAL dirspec[,...]
   ```

   where *dirspec* is the full specification of a directory containing one or more transaction logs. List all the directories that contain any transaction logs other than the transaction log of the node you are removing. You can list the directories in any order.

   In a cluster, use SYSMAN to redefine SYS$JOURNAL clusterwide.
6. If you redefined SYS$JOURNAL in step 5, edit the SYS$MANAGER:SYLOGICALS.COM command procedure to update the SYS$JOURNAL definition.

   If you created node-specific versions of SYLOGICALS.COM, edit all the versions.
7. Archive the transaction log.
8. Shut down the node.
9. Restart the software that uses DECdtm services.
10. Reconfigure the cluster to remove the node.

    For information about how to reconfigure a cluster, refer to *VSI OpenVMS Cluster Systems Manual*.

Example

This example shows how to remove the node BLUE. In this example, the cluster members and the locations of their transaction logs are as follows:

Node	Directory Containing Log
BLUE	DISK$LOG3:[LOGFILES]
RED	DISK$LOG2:[LOGFILES]
WHITE	DISK$LOG4:[LOGFILES]

None of the nodes has a node--specific version of the SYLOGICALS.COM command procedure.

Log in to node BLUE.

Stop all the software that uses DECdtm services. Then find out if BLUE's transaction log contains any active transactions:

```
$ RUN SYS$SYSTEM:LMCP
LMCP> DUMP/ACTIVE SYSTEM$BLUE.LM$JOURNAL
Dump of log file DISK$LOG3:[LOGFILES]SYSTEM$BLUE.LM$JOURNAL
   .
   .
   .
Total of 0 transactions active, 0 prepared and 0 committed.
LMCP> EXIT
```

Redefine SYS$JOURNAL:

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> SET ENVIRONMENT/CLUSTER
SYSMAN> DO DEFINE/SYSTEM/EXECUTIVE_MODE SYS$JOURNAL -
_SYSMAN> DISK$LOG2:[LOGFILES], DISK$LOG4:[LOGFILES]
SYSMAN> EXIT
```

Edit the SYS$MANAGER:SYLOGICALS.COM command procedure to update the SYS$JOURNAL definition.

Archive BLUE's transaction log. Then shut down node BLUE:

```
$ @SYS$SYSTEM:SHUTDOWN.COM
   .
   .
   .
Should an automatic system reboot be performed [NO]? NO
```

Restart the software that uses DECdtm services. Then reconfigure the cluster:

```
$ @SYS$STARTUP:CLUSTER_CONFIG.COM
       Cluster Configuration Procedure

            1. ADD a node to a cluster.
            2. REMOVE a node from the cluster.
            3. CHANGE a cluster member's characteristics.
            4. CREATE a duplicate system disk for BLUE.

    Enter choice [1]: 2
   .
   .
   .
 Updating network database...
 The configuration procedure has completed successfully
```

13.12. Disabling DECdtm Services

By default, DECdtm services start automatically when you boot the computer. The DECdtm process, TP_SERVER, then checks for a transaction log, and continues checking until it finds one.

Disable DECdtm services if you do not use, and do not plan to use, any software that uses DECdtm services. This saves memory and CPU time.

In an OpenVMS Cluster, disable DECdtm services on all the nodes in the cluster.

How to Perform This Task

1. For each node:

   a. Log in to the node.

   b. Stop the TP_SERVER process using LMCP's CLOSE LOG command:

      ```
      $ RUN SYS$SYSTEM:LMCP
      LMCP> CLOSE LOG
      ```

      The CLOSE LOG command stops the TP_SERVER process, providing no software is using DECdtm services.

      If the CLOSE LOG command fails, do not continue this task. If you have already stopped the TP_SERVER process on other nodes in a cluster system, restart the process using the SYS $STARTUP:DECDTM$STARTUP.COM command procedure.

2. Add the following line to the SYS$MANAGER:SYLOGICALS.COM command procedure:

   ```
   $ !
   $ DEFINE/SYSTEM/EXECUTIVE_MODE SYS$DECDTM_INHIBIT yes
   $ !
   ```

   If you created node-specific versions of SYLOGICALS.COM, edit all the versions.

   This stops the TP_SERVER process being created the next time you boot the system.

13.13. Enabling DECdtm Services

Enable DECdtm services only if you have previously disabled them and you now want to run software that uses DECdtm services.

How to Perform This Task

1. Deassign the logical name SYS$DECDTM_INHIBIT:

   ```
   $ DEASSIGN/SYSTEM/EXECUTIVE_MODE SYS$DECDTM_INHIBIT
   ```

   In an OpenVMS Cluster, use SYSMAN to deassign SYS$DECDTM_INHIBIT clusterwide.

2. Start up the DECdtm services process, TP_SERVER:

   ```
   $ @SYS$STARTUP:DECDTM$STARTUP.COM
   ```

   In an OpenVMS Cluster, use SYSMAN to start up the TP_SERVER process clusterwide.

3. Edit the SYS$MANAGER:SYLOGICALS.COM command procedure to delete the SYS $DECDTM_INHIBIT definition. This ensures that DECdtm services start automatically when you boot the system.

Example

This example shows how to enable DECdtm services in a cluster environment.

Deassign SYS$DECDTM_INHIBIT, then start up the TP_SERVER process.

```
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> SET ENVIRONMENT/CLUSTER
SYSMAN> DO DEASSIGN/SYSTEM/EXECUTIVE_MODE SYS$DECDTM_INHIBIT
SYSMAN> DO @SYS$STARTUP.DECDTM$STARTUP.COM
SYSMAN> EXIT
```

Edit the SYS$MANAGER:SYLOGICALS.COM command procedure to delete the SYS $DECDTM_INHIBIT definition.

13.14. Using the XA Gateway (Alpha Only)

DECdtm/XA provides support for coordinating and managing transactions that use XA. By using an XA Gateway, DECdtm/XA can join other Resource Managers (RM) in transactions that are managed by another Transaction Manager (TM). This section describes how to configure and use DECdtm XA Gateway support.

Note

In this chapter, the term XA Specification refers to Distributed Transaction Processing: The XA Specification.

To use DECdtm/XA and ensure proper startup and shutdown of DECdtm/XA services, the following command files must be invoked:

- SYS$STARTUP:DDTM$XA_STARTUP.COM
- SYS$STARTUP:DDTM$XA_SHUTDOWN.COM

Add the command @SYS$STARTUP:DDTM$XA_STARTUP.COM to the startup database or to the command file SYS$MANAGER:SYSTARTUP_VMS.COM.

Add the command @SYS$STARTUP:DDTM$XA_SHUTDOWN.COM to the command file SYS $MANAGER:SYSHUTDWN.COM.

Perform the following steps to verify that DECdtm XA services are operating properly:

1. Use the XGCP utility to create a gateway log file with the same name as the local OpenVMS node. See *Section 13.14.1, "Gateway Configuration"* and the *VSI OpenVMS System Management Utilities Reference Manual* for details.

2. Run SYS$TEST:DECDTM_XG_IVP.EXE.

3. Use the XGCP utility to stop and restart the gateway server. This step is essential if you choose to configure the gateway with a name different than that of the local OpenVMS node. For more information on the XGCP utility, see *VSI OpenVMS System Management Utilities Reference Manual: M-Z*.

13.14.1. Gateway Configuration

The XA Gateway is configured into each transaction processing (TP) process as an XA-compliant resource manager. The XA Gateway handles XA calls from the XA transaction manager (TM) and maps

them into calls to DECdtm system services. This allows DECdtm to send the appropriate events to any DECdtm compliant Resource Manager (RM) used in a TP process.

The operation of the XA Gateway is transparent to the RM; DECdtm RMs do not need any modification to be used with the XA Gateway.

The XA Gateway uses a log file to record the mapping between XA transactions and DECdtm transactions. The log file is managed by the gateway server process DDTM$XG_SERVER.

Create the gateway log file with the XGCP utility (see the *VSI OpenVMS System Management Utilities Reference Manual*). The size of the gateway log file depends on the number of concurrently active transactions. Up to 600 bytes are required for each active transaction, depending on the size of the transaction ID (TID) used by the XA TM. The gateway log file expands automatically when required.

The gateway log file resides in the directory specified by the logical name SYS$JOURNAL and has a name of the form SYSTEM$name.DDTM$XG_JOURNAL. For optimum performance, move each gateway log file and each DECdtm log file to a separate physical device, and define SYS$JOURNAL as a search list for the set of physical devices.

The XA Gateway requires an association on each OpenVMS Cluster node between an XA transaction manager and the XA Gateway log file. This association is managed by specifying a gateway name as follows:

- Use the XGCP utility to create a gateway log file with the gateway name (see the *VSI OpenVMS System Management Utilities Reference Manual*).
- The gateway name is specified in the xa_open information string when the Gateway RM is configured in applications run under the control of an XA TM. (XA RM configuration is described in *OpenVMS Programming Concepts: Volume II*.)
- The first XA application run by the XA TM binds the gateway name to the local node of the OpenVMS Cluster. The gateway name remains bound to that node until the gateway server is stopped.

All XA applications that run on the local node must be configured with the same gateway name. XA applications using the same name cannot run on other OpenVMS Cluster nodes. Therefore, you normally define one gateway name and create one gateway log file for each node of an OpenVMS Cluster.

You can change the association of a gateway name and bind the gateway name to a different OpenVMS Cluster node, provided that the node can access the gateway log file. To change the association of a gateway name, perform the following steps:

1. Stop all XA applications on the original node.
2. Use the XGCP utility to stop the gateway server on the original node.
3. Stop all XA applications on the new node.
4. Stop the gateway server on the new node and then restart the gateway server.
5. Run the original XA applications on the new node.

Note

You must take care to protect against the loss of a gateway log file, perhaps by shadowing the device on which it resides. If you create a new log file, or if you use an out-of-date log file, transactions that were

originally recorded as committed may be incorrectly rolled back. This can cause databases to become inconsistent with each other, or inconsistent with reports given to other systems or users.

Gateway log files are not large and it is better never to delete them. If you do delete an unwanted gateway log file, first use the DECdtm XGCP utility to verify that the gateway is not still a participant in any prepared transactions. The gateway participant name is DDTM$XG/name.

The gateway server uses the following system logical names:

- SYS$DECDTM_XG_REQS

  Number of concurrent requests processed by the server, in the range 100 to 100,000. This determines the size of the global section that DDTM$XG uses for communication with the server, and the quotas required by the server. The parameter is specified by defining the logical name SYS $DECDTM_XG_REQS. Changes to the parameter do not take effect until after the server and all client processes have been stopped.

  If the value of this parameter is exceeded during operation, client requests are blocked instead of being processed in parallel.

- SYS$DECDTM_XA_TRANS

  Estimated number of concurrent XA transactions, in the range 1000 to 1,000,000. This determines the size of indexing tables used internally in the server. The parameter is specified by defining the logical name SYS$DECDTM_XA_TRANS. Changes to this parameter do not take effect until after the server has been stopped.

  If the value of this parameter is exceeded during operation, server CPU use will increase. However, the effect is unlikely to be noticeable until the value of this parameter is exceeded by a factor of 10 or more.

Chapter 14. Managing Special Processing Environments

The OpenVMS operating system supports the following special environments:

- Symmetric multiprocessing
- Vector processing (available only on certain CPU models)

This chapter describes how to set up and manage these special processing environments.

Information Provided in This Chapter

This chapter describes the following tasks:

Task	Section
Creating a multiprocessing environment	*Section 14.2.1, "Creating a Multiprocessing Environment"*
Monitoring a multiprocessing environment	*Section 14.2.2, "Monitoring a Multiprocessing Environment"*
[dag]Loading the vector processing support code	*Section 14.4.1, "Loading the Vector Processing Support Code (VAX Only)"*
[dag]Configuring a vector processing system	*Section 14.4.2, "Configuring a Vector Processing System (VAX Only)"*
[dag]Managing vector processes	*Section 14.4.3, "Managing Vector Processes (VAX Only)"*
[dag]Restricting access to the vector processor with ACLs	*Section 14.4.4, "Restricting Access to the Vector Processor by Using ACLs (VAX Only)"*
[dag]Obtaining information about a vector processing system	*Section 14.4.5, "Obtaining Information About a Vector Processing*

Task	Section
	System (VAX Only)"
[dag]Loading the VAX Vector Instruction Emulation facility (VVIEF)	*Section 14.4.6, "Loading the VAX Vector Instruction Emulation Facility (VVIEF) (VAX Only)"*

[dag]VAX specific

This chapter explains the following concepts:

Concept	Section
Symmetric multiprocessing	*Section 14.1, "Understanding Multiprocessing"*
Primary and secondary processors	*Section 14.1.1, "Primary and Secondary Processors"*
Available and active sets	*Section 14.1.2, "Available and Active Sets"*
Vector processing	*Section 14.3, "Understanding Vector Processing"*
[dag]VAX support for vector processing	*Section 14.3.1, "VAX Support for Vector Processing (VAX Only)"*
[dag]The VAX Vector Instruction Emulation facility (VVIEF)	*Section 14.3.2, "VAX Vector Instruction Emulation Facility (VAX Only)"*

[dag]VAX specific

14.1. Understanding Multiprocessing

A multiprocessing system consists of two or more CPUs that address a common pool of memory and that are capable of executing instructions simultaneously.

The OpenVMS operating system supports a tightly coupled, symmetric multiprocessing (SMP) system. In a tightly coupled SMP system, all processors execute a single copy of the operating system and have equal access to all operating system code and system resources. OpenVMS SMP dynamically selects the CPU where a process will run based on process priority.

A multiprocessing system can function as an isolated entity, a node in a network, or a member of an OpenVMS Cluster environment. Multiprocessing and uniprocessing systems run the same operating

system, although multiprocessing can be enabled only on selected VAX and Alpha processors. All processors in a multiprocessing environment must be at the same hardware and firmware level to guarantee that a given processor is capable of resuming the execution thread of a process that had been executing previously on another processor in the system.

14.1.1. Primary and Secondary Processors

In a multiprocessing system, one processor has the responsibility of starting other processors in the system. The **primary processor** is that processor in the system that is either logically or physically attached to the console device. As such, it is the processor that is the target of the console commands that boot the multiprocessing system. In this role, only the primary processor performs the initialization activities that define the operating system environment and prepare memory for the entire system. In addition, the primary processor serves as the system timekeeper, maintaining the system time and monitoring the timer queue for the expiration of its elements. In this sense, all processors in a multiprocessing system that do *not* have these responsibilities are known as **secondary processors**.

14.1.2. Available and Active Sets

An **available set** is made up of the processors that have passed the system's power-on hardware diagnostics and may or may not be actively involved in the system. Together, the primary and the secondary processors comprise the multiprocessing system's available set.

The **active set** is the subset of the VAX or Alpha system's processors that have passed power-on diagnostics and are actively participating in system operations. The operating system identifies each processor in these sets by its CPU ID, a value prevalent in the syntax and displays of certain DCL and utility commands.

14.1.3. Processor Capabilities

The processors in a multiprocessing system offer certain capabilities to the processes executing in the system. The following capabilities are supported:

- Primary
- Quorum
- Run
- Vector (VAX Only)

In addition, mechanisms exist to add and subtract other capabilities.

The Run capability affects CPU starting and stopping operations.

14.2. Managing SMP Environments

Managing symmetric multiprocessing systems (SMP) involves creating and monitoring a multiprocessing environment.

14.2.1. Creating a Multiprocessing Environment

You can control the membership and character of a multiprocessing system at boot time by setting system parameters designed for these purposes. Among the system parameters that manage a multiprocessing system are the following parameters:

Parameter	Function
MULTIPROCESSING	Determines which synchronization image is loaded into the operating system at boot time
SMP_CPUS	Determines which processors are brought into the multiprocessing environment from the available set at boot time

For more information about these and other system parameters, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

You can add an available processor to the active set at boot time, or you can add it later using the DCL command START/CPU. The DCL command STOP/CPU removes a processor from the active set.

SMP Extension License

On Alpha systems, you must register the SMP Extension License if you have an SMP system. This license upgrades the Operating System Base License and all Interactive User licenses to the matching multiprocessing level of your system.

Because the SMP Extension License grants all the rights the existing Base and User licenses provide at the uniprocessing level, reinstalling those licenses when you upgrade to a multiprocessing system is unnecessary. When your system is upgraded to a new multiprocessing level, add an SMP Extension License to your existing license.

14.2.2. Monitoring a Multiprocessing Environment

Several operating system features provide special information about the character, capabilities, and status of a multiprocessor system. They include the DCL command SHOW CPU and the Monitor utility.

Obtaining Information About a Multiprocessor Configuration

The SHOW CPU command displays three levels of information describing the configuration and status of a multiprocessing system:

Level	Command Example	Display Contents
Summary	SHOW CPU	Indicates which processor is primary, which processors are configured, and which processors are active; displays the minimum revision levels for processors in the system and the setting of the MULTIPROCESSING system parameter; and indicates whether multiprocessing is enabled.
Brief	SHOW CPU /BRIEF	Produces information from the summary display; lists the current CPU state and the current process (if any) for each configured processor.
Full	SHOW CPU /FULL	Produces information from the summary display and displays additional information; lists the current CPU state, current process (if any), revision levels, and capabilities for each configured processor; indicates which processes can be executed only on certain processors.

For more information about the DCL commands relating to SMP, refer to the *VSI OpenVMS DCL Dictionary*; for information about the Monitor utility, refer to the *VSI OpenVMS System Management Utilities Reference Manual*.

14.3. Understanding Vector Processing

A single data item, having one value, is known as a **scalar**. A group of related scalar values, or elements, all of the same data type, is known as a **vector**.

Traditional (scalar) computers operate only on scalar values, and must process vector elements sequentially. Vector computers, on the other hand, recognize vectors as native data structures and can operate on an entire vector with a single vector instruction. Because this type of processing involves the concurrent execution of multiple arithmetic or logical operations, a vector computer can routinely process a vector four to five times faster than a traditional computer can using only scalar instructions.

Vector processors gain a further speed advantage over scalar processors by their use of special hardware techniques designed for the fast processing of streams of data. These techniques include data pipelining, chaining, and other forms of hardware parallelism in memory and in arithmetic and logical functional units. Pipelined functional units allow the vector processor to overlap the execution of successive computations with previous computations.

14.3.1. VAX Support for Vector Processing (VAX Only)

The VAX vector architecture includes sixteen 64-bit vector registers (V0 through V15), each containing 64 elements; vector control registers, including the vector count register (VCR), vector length register (VLR), and vector mask register (VMR); vector functional units; and a set of vector instructions. VAX vector instructions transfer data between the vector registers and memory, perform integer and floating-point arithmetic, and execute processor control functions. A more detailed description of the VAX vector architecture, vector registers, and vector instructions appears in the *VAX MACRO and Instruction Set Reference Manual*.

Those VAX systems that comply with the VAX vector architecture are known as **vector-capable systems**.

A VAX vector processing system configuration includes one or more integrated scalar-vector processor pairs, or **vector-present processors**. Such a configuration can be symmetric, including a vector coprocessor for each scalar, or asymmetric, incorporating additional scalar-only processors. Depending upon the model of the VAX vector processing system, the scalar and vector CPUs of vector-present processors can be either a single, integral physical module or separate, physically independent modules. In either case the scalar and vector CPUs are logically integrated, sharing the same memory and transferring data over a dedicated, high-speed internal path.

Like VAX scalar processing systems, a VAX vector processing system can participate as a member of a VAXcluster or a node in a network, or be run as a standalone system.

14.3.2. VAX Vector Instruction Emulation Facility (VAX Only)

The VAX Vector Instruction Emulation Facility (VVIEF) is a standard feature of the OpenVMS operating system that allows vectorized applications to be written and debugged in a VAX system in which vector processors are not available. VVIEF emulates the VAX vector processing environment, including the nonprivileged VAX vector instructions and the vector system services. Use of VVIEF is restricted to user mode code.

VVIEF is strictly a program development tool, and *not* a run-time replacement for vector hardware. Vectorizing applications to run under VVIEF offers no performance benefit; vectorized applications running under VVIEF execute more slowly than their scalar counterparts.

The operating system supplies the VVIEF bootstrap code as an executive loadable image. Note that, in the presence of OpenVMS vector support code, VVIEF remains inactive. Although it is possible to prevent the loading of vector support code in a vector-present system (see *Section 14.4.1, "Loading the Vector Processing Support Code (VAX Only)"*) and activate VVIEF, there are few benefits.

See *Section 14.4.6, "Loading the VAX Vector Instruction Emulation Facility (VVIEF) (VAX Only)"* for additional information about loading and unloading VVIEF.

14.4. Managing the Vector Processing Environment (VAX Only)

The following sections describe tasks for managing a vector processing system.

14.4.1. Loading the Vector Processing Support Code (VAX Only)

By default, in a VAX vector processing system, the system automatically loads the vector processing support code at boot time. You can override the default behavior by setting the static system parameter VECTOR_PROC as described in *Table 14.1, "Settings of VECTOR_PROC System Parameter(VAX Only)"*.

Table 14.1. Settings of VECTOR_PROC System Parameter (VAX Only)

Value	Result
0	Do not load the vector processing support code, regardless of the system configuration.
1	Load the vector processing support code if at least one vector-present processor exists. This is the default value.
2	Load the vector processing support code if the system is vector-capable. This setting is most useful for a system in which processors have separate power supplies. With this setting, you can reconfigure a vector processor into the system without rebooting the operating system.

14.4.2. Configuring a Vector Processing System (VAX Only)

You can add a vector-present processor to or remove it from a multiprocessing configuration at boot time by using the system parameter SMP_CPUS, or at run time by using the DCL commands START/CPU and STOP/CPU. Note that the operating system treats the scalar and vector CPU components of a vector-present processor as a single processor, starting them and stopping them together.

At boot time, the setting of the system parameter SMP_CPUS identifies which secondary processors in a multiprocessing system are to be configured, including those processors that are vector present. (The operating system always configures the primary processor.) The default value of –1 boots all available processors, scalar and vector-present alike, into the configuration. (Refer to the *VSI OpenVMS System*

Management Utilities Reference Manual for additional information about this parameter.) Note that, prior to starting a vector-present processor, you should ensure that the vector processing support code (see *Section 14.4.1, "Loading the Vector Processing Support Code (VAX Only)"*) is loaded at boot time. Otherwise, processes will be able to use only the scalar CPU component of the vector-present processor.

To bring secondary processors into a running multiprocessing system, use the DCL command START/CPU. To remove secondary processors from the system, use the STOP/CPU commands. Again, you must ensure that the vector processing support code has been loaded at boot time for the vector CPU component of vector-present processors started in this way to be used.

Note, however, that a STOP/CPU command fails and generates a message if it would result in the removal of a vector-present processor that is the sole provider of the vector capability for currently active vector consumers. In extreme cases, such as the removal of a processor for repair, you can override this behavior by issuing the command STOP/CPU/OVERRIDE. This command stops the processor, despite stranding processes.

When a STOP/CPU/OVERRIDE command is issued for a vector-present processor, or when a vector-present processor fails, the operating system puts all stranded vector consumers into a CPU-capability-wait (RSN$_CPUCAP) state until a vector-present processor is returned to the configuration. To any other process that subsequently issue a vector instruction (including a marginal vector consumer), the system returns a “requested CPU not active” message (CPUNOTACT).

Refer to the *VSI OpenVMS DCL Dictionary* for additional information about the START/CPU and STOP/CPU commands.

14.4.3. Managing Vector Processes (VAX Only)

The operating system scheduling algorithms automatically distribute vector and scalar processing resources among vector consumers, marginal vector consumers, and scalar consumers. However, VAX vector processing configurations vary in two important ways:

- The amount of vector processing activity the configuration must accommodate
- The number of vector-present processors that are available in the configuration to service vector processing needs

In a configuration that has more vector consumers in a system than scalar-vector processor pairs to service them, vector consumers share vector-present processors according to process priority. At a given priority, the system schedules vector consumers on a vector-present processor in a round-robin fashion. Each time the system must schedule a new vector consumer on a vector-present processor, it must save the vector context of the current vector consumer in memory and restore the vector context of the new vector consumer from memory. When such “slow” vector context switches occur too frequently, a significant portion of the processing time is spent on vector context switches relative to actual computation.

Systems that have heavy vector processing needs should be adequately configured to accommodate those needs. However, some mechanisms are available for tuning the performance of an existing configuration.

14.4.3.1. Adjusting System Resources and Process Quotas (VAX Only)

Systems in which several vector consumers are active simultaneously may experience increased paging activity as processes share the available memory. To reduce process paging, you may need to use the

Authorize utility (AUTHORIZE) to adjust the working set limits and quotas of the processes running vectorized applications. (Refer to the AUTHORIZE section of the *VSI OpenVMS System Management Utilities Reference Manual* for additional information.) An increase of the process maximum working set size (system parameter WSMAX) may also be necessary. Additionally, a vectorized application may use the Lock Pages in Working Set system service ($LKWSET) to enhance its own performance.

The system allots to each vector consumer 8KB of system nonpaged dynamic memory in which the operating system stores vector context information. Depending upon how many vector consumers may be active in the system simultaneously, you may need to adjust the system parameter NPAGEDYN. The DCL command SHOW MEMORY/POOL/FULL displays the current size of non paged pool in bytes.

To obtain optimal performance of a VAX vector processing system, you should take some care in setting up generic batch queues that avoid saturating the system's vector resources. If a queue contains more active vectorized batch jobs than vector-present processors in the system, a significant portion of the processing time will be spent on vector context switches.

The recommended means for dispatching vectorized batch jobs to a VAX vector processing system is to set up a separate queue (for instance, VECTOR_BATCH) with a job limit equal to the number of vector-present processors in the system. When submitting vectorized batch jobs, users should be encouraged to submit them to this generic vector-processing batch queue.

14.4.3.2. Distributing Scalar and Vector Resources Among Processes (VAX Only)

As a vector consumer, a process must be scheduled only on a vector-present processor. If the image the process is executing issues only scalar instructions for a period of time, and it must share the scalar-vector processor pair with other vector consumers, its inability to run on an available scalar processor could hamper its performance and the overall performance of the system.

By default, the operating system assumes that if a vector consumer has not issued a vector instruction for a certain period of time, it is unlikely that it will issue a vector instruction in the near future. The system relinquishes this process's need for the vector capability, classifying it as a marginal vector consumer.

In an asymmetric vector-processing configuration, detection of marginal vector consumers achieves the following desirable effects:

- Because a marginal vector consumer is eligible to run on a larger set of processors, its response time will improve.
- The scheduling of marginal vector consumers on scalar processors reduces the contention for vector-present processors.
- Because vector consumers issuing vector instructions are more likely to be scheduled on vector-present processors, the vector CPU is more efficiently used.

Use the VECTOR_MARGIN system parameter to establish the interval of time at which the system checks the status of all vector consumers. The VECTOR_MARGIN parameter accepts an integer value between 1 and $FFFFFFFF_{16}$. This value represents a number of consecutive process quanta (as determined by the system parameter QUANTUM). If the process has not issued any vector instructions in the specified number of quanta, the system declares it a marginal vector consumer.

The default value of the VECTOR_MARGIN parameter is 200_{10}.

14.4.4. Restricting Access to the Vector Processor by Using ACLs (VAX Only)

A vector **capability** is a software abstract by which the operating system makes the services of the vector processor available to users. You can restrict the use of the vector processor to users holding a particular identifier by associating an access control list (ACL) with the vector capability object.

For example, a university might limit use of the vector processor to faculty and students in an image processing course, or a service bureau might charge users for access to the vector capability, time spent onthe vector processor, or both.

Use the DCL command SET SECURITY/ACL in the following format to establish access control entries (ACEs) on a vector capability:

```
SET SECURITY /CLASS=CAPABILITY /ACL=(ace[, ...]) VECTOR
```

The following DCL command displays the ACL on the vector capability:

```
$ SHOW SECURITY /CLASS=CAPABILITY VECTOR
```

Note that the ACL is on the vector capability, not on the use of any or all vector-present processors in the system. The operating system will still schedule processes without permission to use the vector capability on a vector-present processor. However, these processors will be able to use only the scalar CPU component of the processor, and cannot execute vector instructions. Likewise, because the ACL is on the vector capability and not on a vector-present processor, you cannot establish an ACL to force long-running jobs to a specific processor.

For additional information about the SET SECURITY and SHOW SECURITY commands, refer to the *VSI OpenVMS DCL Dictionary*.

14.4.5. Obtaining Information About a Vector Processing System (VAX Only)

You can obtain information about the status of the vector processing system and the use of the system by individual processes through various means, including:

- The DCL lexical functions F$GETJPI and F$GETSYI
- The DCL command SHOW CPU
- The DCL commands SHOW PROCESS and LOGOUT/FULL
- The Accounting utility
- The Monitor utility

14.4.5.1. DCL Lexical Functions F$GETJPI and F$GETSYI (VAX Only)

The DCL lexical function F$GETJPI accepts the following items and returns the corresponding information regarding the vector status of a specified process:

Item	Return Type	Information Returned
FAST_VP_SWITCH	Integer	Number of times this process has issued a vector instruction that resulted in an inactive vector processor being enabled without the expense of a vector context switch
SLOW_VP_SWITCH	Integer	Number of times this process has issued a vector instruction that resulted in an inactive vector processor being enabled with a full vector context switch
VP_CONSUMER	Boolean	Flag indicating whether the process is a vector consumer
VP_CPUTIM	Integer	Total amount of time the process has accumulated as a vector consumer

The DCL lexical function F$GETSYI accepts the following items and returns the corresponding information regarding the status of the vector processing system:

Item	Return Type	Information Returned
VECTOR_EMULATOR	Integer	Flag indicating the presence of the VAX Vector Instruction Emulation facility (VVIEF) in the system
VP_MASK	Integer	Mask indicating which processors in the system have vector coprocessor
VP_NUMBER	Integer	Number of vector processors in the system

Refer to the *VSI OpenVMS DCL Dictionary* for additional information about the DCL lexicals F$GETJPI and F$GETSYI.

14.4.5.2. SHOW CPU/FULL Command (VAX Only)

The SHOW CPU/FULL command lists the capabilities of the specified CPU. Issue this command to determine the presence of the vector capability in the system prior to executing a STOP/CPU command.

Refer to the *VSI OpenVMS DCL Dictionary* for additional information about the SHOW CPU command.

14.4.5.3. SHOW PROCESS and LOGOUT/FULL Commands (VAX Only)

If the target process has accrued any time as a vector consumer scheduled on a vector-present processor, the DCL commands SHOW PROCESS and LOGOUT/FULL display the elapsed vector CPU time and the charged vector CPU time, respectively.

To accumulate vector CPU time, a process must be a vector consumer (that is, require the system vector capability) and be scheduled on a vector-present processor. The operating system still charges the vector consumer vector CPU time, even if, when scheduled on the vector-present processor, it does not actually use the vector CPU. Note that, because scalar consumers and marginal vector consumers do not use the vector CPU, they do not accrue vector CPU time, even when scheduled on a vector-present processor.

Refer to the *VSI OpenVMS DCL Dictionary* for additional information about the SHOW PROCESS and LOGOUT commands.

14.4.6. Loading the VAX Vector Instruction Emulation Facility (VVIEF) (VAX Only)

The VAX Vector Instruction Emulation Facility (VVIEF) is a standard operating system feature that allows vectorized applications to be written and debugged in a VAX system in which vector processors are not available. VVIEF is intended strictly as a program development tool, and *not* as a run-time replacement for vector hardware. Vectorizing applications to run under VVIEF offers no performance benefit; vectorized applications running under VVIEF will execute more slowly than their scalar counterparts.

To cause the system to load VVIEF at the *next* system boot and at each subsequent system boot, invoke the command procedure SYS$UPDATE:VVIEF$INSTAL.COM. To unload VVIEF, invoke the command procedure SYS$UPDATE: VVIEF$DEINSTAL.COM and reboot the system.

You can determine the presence or absence of VVIEF in a system by issuing the following DCL commands:

```
$ X = F$GETSYI("VECTOR_EMULATOR")
$ SHOW SYMBOL X
X = 1   Hex = 00000001  Octal = 0000000001
```

A return value of 1 indicates the presence of VVIEF; a value of 0 indicates its absence.

Note that, although VVIEF may be loaded into the system, in the presence of vector support code, it remains inactive. Although it is possible to prevent the loading of vector processing support code in a vector-present system (see *Section 14.4.1, "Loading the Vector Processing Support Code (VAX Only)"*) and activate VVIEF, there are few benefits. Should the only vector-present processor in the system fail, the execution of preempted vectorized applications will not resume under VVIEF.

Appendix A. Files–11 Disk Structure

This appendix explains disk terminology and disk concepts. It also describes reserved files, points out those files used by the Analyze/Disk_Structure utility (ANALYZE/DISK_STRUCTURE), and compares Files-11 On-Disk Structure (ODS) Level 1 with Files-11 ODS Levels 2 and 5.

A.1. Disk Concepts

This section defines terms related to both the physical and the logical organization of disks.

Logical Organization of a Disk

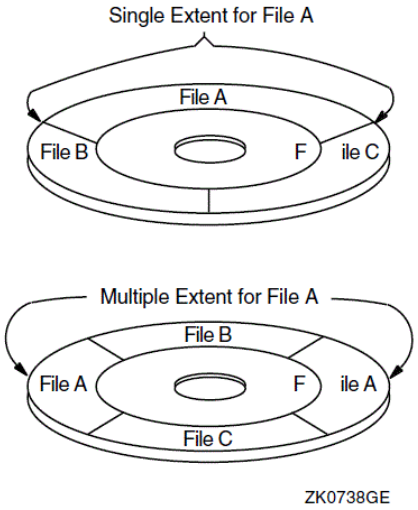
The smallest addressable unit of information on a disk is a block. Files--11 On-Disk Structures define a block to consist of 512 8-bit bytes. Blocks can be treated as units for transfer between a Files-11 disk volume and memory. Files--11 ODS, however, views a disk as an array of blocks, and is generally not concerned with individual blocks.

Blocks are logically grouped into clusters, which are the basic units by which disk space is allocated. You determine the number of blocks in a cluster when a given disk, known as a volume, is first prepared for use (initialized). Cluster sizes vary for different media types. The smaller cluster sizes in the range are usually more practical. In general, a disk with a relatively small number of blocks is given a smaller cluster size, while larger disks are given larger cluster sizes to minimize the overhead for disk space allocation.

Contiguous clusters allocated to a particular file are called **extents**. An extent can contain all or part of a file. If enough contiguous area is available on the disk, the entire file is allocated as a single extent. Sometimes, however, not enough contiguous area is available to hold the entire file, or, when you create a file initially, you might not want to reserve the entire required amount of space. When the file is eventually extended, it is unlikely that the adjacent clusters will still be unallocated. If the adjacent clusters are already allocated to another file, the extension does not occur contiguously.

If a file is divided into two or more parts, each part is an extent. Thus, a file can consist of multiple extents located in separate areas on the disk, as shown in *Figure A.1, "File Extents"*. Note that the file extensions are done automatically.

Figure A.1. File Extents



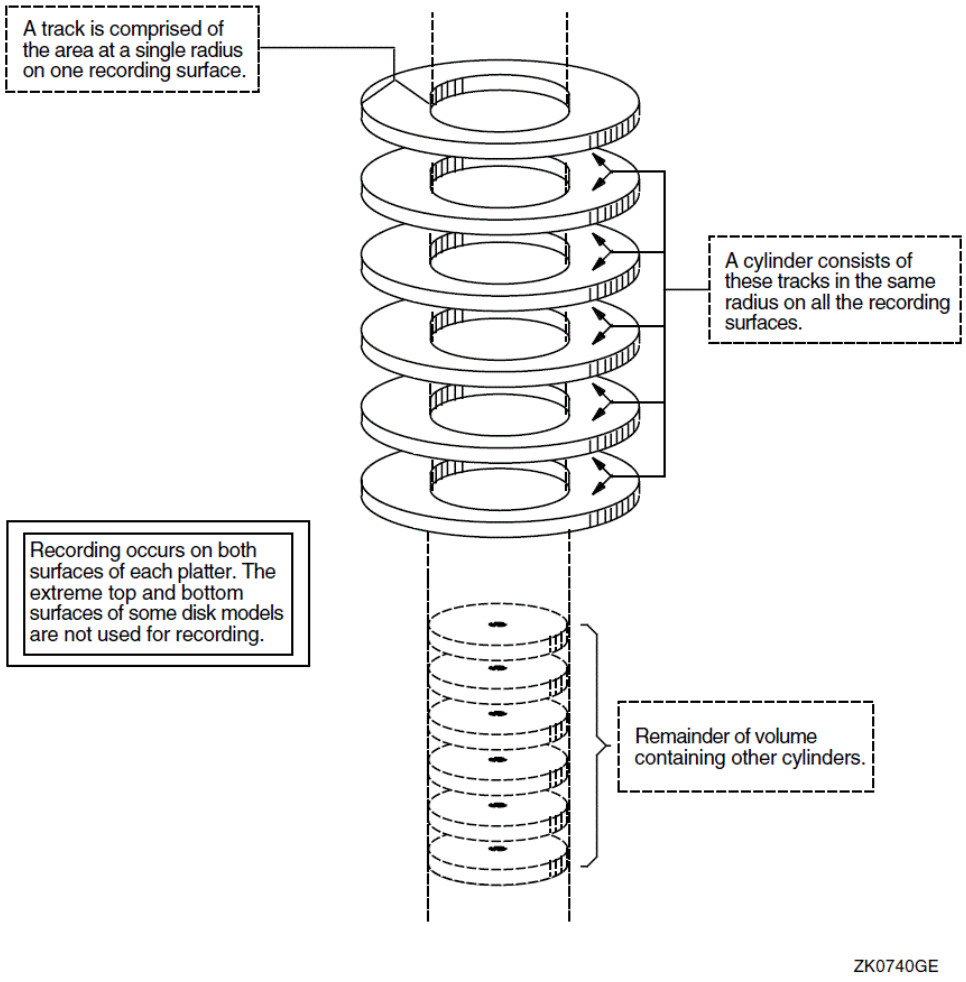
A.1.1. Physical Organization of a Disk

The smallest unit discernible to the Files-11 structure is the **sector**; for most Files-11 disks, a sector is equivalent to a block, which is 512 bytes. Other basic terms related to disks are **track** and **cylinder**. A track is the collection of sectors (or blocks, on Files-11 structures) at a single radius on one recording surface of a disk. It is accessible to a given read/write head position on the disk device. A cylinder consists of all track sat the same radius on all recording surfaces of a disk.

Because access to any of the blocks in a given cylinder does not require any movement of the disk's read/write heads, it is generally advantageous to keep related data blocks in the same cylinder. For this reason, when choosing a cluster size for a large-capacity disk, you should usually select a cluster size that divides evenly into the cylinder size.

Figure A.2, "Tracks and Cylinders" is a graphic representation of disk tracks and cylinders.

Figure A.2. Tracks and Cylinders



A.2. Files–11 Structure

The Files-11 structure creates a set of non deletable reserved files when a volume or volume set is initialized. These files control the organization of a Files-11 disk. A Files-11 structure resides on a volume, which is a physical medium such as a disk pack. A Files-11 volume is an ordered set of 512-byte blocks. The blocks are numbered consecutively from 0 to `n-1`; the value of `n-1` is the size of the disk in blocks.

A.2.1. File Identification (FID)

Each file on a Files-11 disk is identified by a unique, system-assigned file identification (FID) and can have a user-assigned alphanumeric name. The primary function of a Files-11 directory is to associate the user-assigned alphanumeric name of each file with the unique FID of the file. This association ensures that files present on a volume are retrievable by name.

The FID of a file consists of a set of three numbers. The first is the *file number* (NUM). The file system uses this number as an offset into the index file (reserved file INDEXF.SYS), which stores information for all files on a volume.

The second part of the FID is the *file sequence number* (SEQ), which represents the number of times a particular file number has been used. File numbers are allocated and deallocated as files are created and deleted. As a result, the file number alone cannot uniquely identify the file. By incrementing the

sequence number each time a file number is used, the file system ensures that each file has a unique identification in INDEXF.SYS.

The third number in the FID is the *relative volume number* (RVN). This number indicates the volume (of a volume set) on which the file resides (ODS–2 only). If the volume set consists of a single volume, the RVN of all files on that volume is 1.

A.2.2. ODS Directory Hierarchies

2 structure is a multilevel directory hierarchy. The top level of the directory structure is the master file directory (MFD). The MFD of a volume is always named [000000]. The MFD contains all top-level directories, including itself, and reserved files.

A directory is a file that contains other files. A file contained in a directory can also be a directory and contain other files. By nesting directories, users can construct directory hierarchies up to nine levels deep (including the master file directory).

In a volume set, the MFD for all of the user directories on the volume set is located on relative volume 1. The entries of this MFD point to directories located on any volume in the set. These directories in turn point to files and subdirectories on any volume in the set. The MFD of any remaining volume in the set includes only the names of the reserved files for that volume.

On VAX systems, the Files-11 ODS–1 structure supports a two-level directory hierarchy. Each user identification code (UIC) is associated with a user file directory (UFD). Each UFD is included in the MFD of the volume.

A.3. Reserved Files

This section describes the reserved files that Files–11 uses. Note that all reserved files have constant FIDs.

This section also points out the files ANALYZE/DISK_STRUCTURE uses. ANALYZE/DISK_STRUCTURE makes an in-memory copy of what these files should look like and compares it with the current version. Rebuilds specific Files–11 reserved files and compares these files with their old versions. The utility reports and repairs (if you specify the /REPAIR qualifier) any discrepancies found during these comparisons.

Table A.1, "Reserved Files" shows the reserved files used by Files–11 Levels 1, 2, and 5, and files used by ANALYZE/DISK_STRUCTURE.

Table A.1. Reserved Files

Reserved File	File Name	[dag]Structure Level 1	Structure Levels 2 and 5	ANALYZE / DISK_ STRUCTURE
Index file	INDEXF.SYS;1	X	X	X
Storage bitmap file	BITMAP.SYS;1	X	X	X
Bad block file	BADBLK.SYS;1	X	X	
Master file directory	000000.DIR;1	X	X	X
Core image file	CORIMG.SYS;1	X	X	
Volume set list file	VOLSET.SYS;1		X	X

Reserved File	File Name	[dag]Structure Level 1	Structure Levels 2 and 5	ANALYZE / DISK_ STRUCTURE
Continuation file	CONTIN.SYS;1		X	
Backup log file	BACKUP.SYS;1		X	
Pending bad block	BADLOG.SYS;1		X	
Quota file	QUOTA.SYS			X
Volume security profile	SECURITY.SYS		X	

[dag]VAX specific

A.3.1. Index File, INDEXF.SYS

Every Files–11 volume has an index file, which is created when the volume is initialized. (You cannot use a disk as a Files-11 disk until it has been initialized with the INITIALIZE command.)

INDEXF.SYS is a large, extendable file made up of several sections. These sections provide the operating system with the information necessary to identify a Files-11 volume, initially access that volume, and locate all the files on that volume (including INDEXF.SYS itself).

Table A.2, "Contents of Files–11 Index File" shows the information that is in INDEXF.SYS. After the table are additional explanations of boot block, home block, and file headers.

Table A.2. Contents of Files–11 Index File

Term	Definition
Boot block	Virtual block 1 of the index file. The boot (or bootstrap) block is almost always mapped to the logical block 0 of the volume. If the volume is a system volume, the boot block contains a boot program that loads the operating system into memory. If the volume is not a system volume, the boot block contains a program that displays the message that the volume is not the system device but a device that contains users' files only.
Home block	Establishes the specific identity of the volume, providing such information as the volume name and protection, the maximum number of files allowed on the volume, and the volume ownership information. The home block is virtual block number 2 of the index file.
Backup home block	A copy of the home block;permits the volume to be used even if the primary home block is destroyed.
Backup index file header	Permits data on the volume to be recovered if the index file header is corrupted; occupies virtual blocks v * 3 + 1 through v * 4, where v is the volume cluster factor.
Index file bitmap	Controls the allocation of file headers and thus the number of files on the volume; contains a bit for each file header allowed on the volume. If the value of a bit for a given file header is 0, a file can be created with this file header. If the value is 1, the file header is already in use.
File headers	Make up the bulk of the index file; contain all the information needed for gaining access to the file. Each file header describes one file on the volume. A file header contains information such as the owner UIC, protection code, creation date and time, and access control lists (ACLs); it also contains a list

Term	Definition
	of extents that make up the file, describing where the file is logically located on the volume. Note that a file header can also be an extension header.
Alternate index file header	Permits recovery of data on the volume if the primary index file header becomes damaged.

A.3.1.1. Boot Block

Block 0 on a system disk is the **boot block**. It contains the location and size of the **primary bootstrap image**, which is used to boot the system. Certain processors, in order to boot, must read this boot block to obtain the location of the bootstrap image. For more details, see *VSI OpenVMS System Manager's Manual, Volume 1: Essentials*.

A.3.1.2. Home Block

The **home block** is normally the next block after the boot block; it identifies the disk as a Files-11 volume. If for some reason the home block cannot be read (physically unusable), an alternative block will be selected for use as the home block. This block provides specific information about the volume and default values for files on the volume. Items in the home block include the following ones:

- The volume name
- Information to locate the remainder of the index file
- The maximum number of files that can be present on the volume at any onetime
- The user identification code (UIC) of the owner of the volume
- Volume protection information (specifies which users can read or write the entire volume)

Files-11 volumes contain several copies of the home block to ensure against accidental destruction of this information and the consequent loss of access to files on the volume.

A.3.1.3. File Headers

Most of the index file consists of **file headers**; each file header describes a portion of a file on the volume. File headers contain information such as the owner UIC, protection code, creation date and time, and access control lists (ACLs). Most importantly, the file header contains a list of extents that make up the file, describing where the file is logically located on the volume. If a file has a large number of extents, multiple file headers may be used to describe them. A file identifier number is associated with each file header.

When you create a file, you normally specify a file name to OpenVMS RMS, which assigns this name to the file on a Files–11 volume. OpenVMS RMS places the file name and file identifier associated with the newly created file into a directory, which contains an entry defining the location for each file. When you access the file, you supply the file name, which supplies a path to the file identifier through the directory entry. The file identifier, in turn, points to the location of the file header, which contains a listing of the extent or extents that locate the actual data.

Because they represent the current state of file storage on a volume, file headers are of particular interest to ANALYZE/DISK_STRUCTURE. Each file on a Files-11 disk (INDEXF.SYS included) is identified and located by a primary header (and extension headers, if required) in INDEXF.SYS.

Each fixed-length header contains both constant and variable-length data. This data is stored in one of the six areas shown in *Table A.3, "Areas of Data in File Headers"*.

Table A.3. Areas of Data in File Headers

Area of Data	Description
Header	This area contains the header identification, the file number and its sequence number, the protection code for the file, and offsets to the other file header areas.
Ident	This area contains the identification and accounting data for the file (for example, the name of the file, its creation date and time, and backup date and time).
Map	This area contains a list of retrieval pointers that map the virtual blocks of the file to the logical blocks of the volume. Each pointer describes one group of consecutively numbered logical blocks that is allocated to the file. Retrieval pointers are arranged in the order of the virtual blocks they represent.
Access control list	An optional area that contains ACL-related information.
Reserved	This area is reserved for use by special applications.
End checksum	The last two bytes of the file header contain a 16-bit additive checksum of the preceding 255 words of the file header. The checksum helps verify that the block is a valid file header.

A set of contiguous clusters is known as an **extent**. The size of an extent varies according to the number of contiguous clusters. For example, assume a file requires 1000 blocks of storage, and the file system finds a set of 800 contiguous blocks and a set of 200 contiguous blocks. The file would then be stored in two extents: one consisting of 800 blocks, the other of 200.

The *primary header* of a file points to the first extent of that file and to as many extents as can be stored in the map area of the primary header. When the number of extents required to contain a file exceeds the map area available in the primary header, or the ACL is too large to fit in the primary header, the file is allocated an *extension header*. Extension headers contain all the constant data of the primary header, as well as the variable data (in the header map area and access control list) that specifies the locations of the extents to which the extension header points.

ANALYZE/DISK_STRUCTURE confirms the validity of a file by working its way down the list of primary and extension headers of the file. During this process, ANALYZE/DISK_STRUCTURE checks the validity of the file header, the chain of pointers to all extension headers, the retrieval pointers in all headers, and the attributes of the file.

A.3.2. Storage Bitmap File, BITMAP.SYS

The storage bitmap file is a contiguous file that the file system uses to keep track of the available space on a volume. This file contains a storage control block (SCB), which consists of summary information intended to optimize the Files–11 space allocation, and the bitmap itself, which lists the availability of individual blocks.

The SCB contains summary information about the volume (cluster factor, volume size, blocking factor, and so forth). Each bit in the bitmap represents an allocatable cluster on the volume. If a bit is set, the corresponding cluster is available for use. If a bit is clear, the cluster is not available.

During normal operation, the operating system moves portions of the bitmap in and out of cache memory. The state of each bit in memory is altered as clusters are allocated and deallocated. BITMAP.SYS is updated when the portion of the bitmap in cache is swapped back to disk. Since a

portion of the bitmap is always in cache, BITMAP.SYS never reflects the current state of allocated clusters on a disk (unless the disk is dismounted or write-locked).

One of the functions of ANALYZE/DISK_STRUCTURE is to build a current version of BITMAP.SYS from data extracted from INDEXF.SYS, so that BITMAP.SYS accurately reflects the status of free clusters on the disk.

A.3.3. Bad Block File, BADBLK.SYS

The bad block file contains all the bad blocks on the volume. The system detects bad disk blocks dynamically and prevents their reuse once the files to which they are allocated have been deleted.

A.3.4. Master File Directory

The MFD is a file that contains reserved files that control the Files-11 volume directory structure. The MFD lists the known files, in addition to any files or directories that the user enters. The master file directory is itself one of the files (000000.DIR;1) listed in the MFD.

Usually, however, the MFD is used to list the reserved files and users' file directories; users seldom enter files into the MFD, even on private volumes. In fact, on a private volume, it is most convenient for users to create a directory that has the same name as their default directory on a system disk. For an explanation of users' file directories and file specifications, refer to the *VSI OpenVMS User's Manual*.

When the Backup utility (BACKUP) creates sequential disk save sets, it stores the save-set file in the MFD.

ANALYZE/DISK_STRUCTURE verifies all files contained in the directory structure by making comparisons to INDEXF.SYS. Any file found in INDEXF.SYS that is not traceable through the directory structure is “lost.” ANALYZE/DISK_STRUCTURE places lost files in the top-level directory SYSLOST.DIR if you specified /REPAIR in the command.

A.3.5. Core Image File, CORIMG.SYS

The core image file is not used by the operating system.

A.3.6. Volume Set List File, VOLSET.SYS

The volume set list file is used only on relative volume 1 of a volume set. The file contains a list of the labels of all the volumes in the set and the name of the volume set.

ANALYZE/DISK_STRUCTURE uses VOLSET.SYS to locate each volume in the set and confirm the attributes of each volume. Since all volume set information is stored in VOLSET.SYS on relative volume 1, ANALYZE/DISK_STRUCTURE ignores VOLSET.SYS on all other volumes.

A.3.7. Continuation File, CONTIN.SYS

The continuation file is used as the extension file identifier when a file crosses from one volume to another volume of a loosely coupled volume set. This file is used for all but the first volume of a sequential disk save set.

A.3.8. Backup Log File, BACKUP.SYS

The backup log file is reserved for future use.

A.3.9. Pending Bad Block Log File, BADLOG.SYS

The pending bad block log file contains a list of suspected bad blocks on the volume that are not listed in the bad block file.

A.3.10. Quota File, QUOTA.SYS

The quota file is a reserved file that is used by the file system to keep track of the disk usage of each UIC on a volume. If you enable disk quota checking for a volume, the records of the file QUOTA.SYS contain all the UICs on the volume. The system constantly updates QUOTA.SYS to reflect the current disk usage, the maximum allowed disk usage, and the permitted overdraft for each UIC.

During the course of its operations, ANALYZE/DISK_STRUCTURE creates a version of QUOTA.SYS in memory that reflects the actual disk usage for each UIC. This version is eventually compared to the disk version of QUOTA.SYS. If ANALYZE/DISK_STRUCTURE detects any disparities in disk usage, ANALYZE/DISK_STRUCTURE notifies you. If you invoked ANALYZE/DISK_STRUCTURE with the /REPAIR qualifier, the disk version of QUOTA.SYS is updated.

A.3.11. Volume Security Profile, SECURITY.SYS

The volume security profile includes the volume owner UIC, the volume system-owner-group-world (SOGW) protection mask, and the volume access control list (ACL).

A.4. Files–11 ODS Level 1 (VAX Only) Versus Levels 2 and 5

On VAX systems, for reasons of performance, reliability, and security, Files–11 ODS Level 2, a compatible superset of ODS Level 1, is the preferred disk structure on the system. At volume initialization time, Structure Level 2 is the default. (Refer to the INITIALIZE command in the *VSI OpenVMS DCL Dictionary*.)

On VAX systems, specify ODS Level 1 only for volumes that must be transportable to RSX–11M, RSX–11D, RSX–11M–PLUS, and IAS systems, as these systems support only that structure level. Additionally, you might be required to handle Structure Level 1 volumes transported to OpenVMS from one of these systems.

Where Structure Level 1 volumes are in use on the system, bear in mind the limitations on them that are shown in *Table A.4, "Limitations on Files–11 Structure Level 1 Volumes"*.

Table A.4. Limitations on Files–11 Structure Level 1 Volumes

Disk	Only Files–11 ODS–2 disks are protected objects.
Directories	No hierarchies of directories and subdirectories, and no ordering of directory entries (that is, the file names) in any way. RSX–11M, RSX–11D, RSX–11M–PLUS, and IAS systems do not support subdirectories and alphabetical directory entries.
Disk quotas	Not supported.
Multivolume files and volume sets	Not supported.
Placement control	Not supported.

Caches	No caching of file header blocks, file identification slots, or extent entries.
System disk	Cannot be a Structure Level 1 volume.
OpenVMS Cluster access	Local access only; cannot be shared across a cluster.
Clustered allocation	Not supported.
Backup home block	Not supported.
Protection code E	E means “extend” for the RSX–11M operating system but is ignored by OpenVMS.
File versions	Limited to 32, 767; version limits are not supported.
Enhanced protection features (for example, access control lists)	Not supported.
Long file names	Not supported.
RMS journalling	Not supported.
RMS execution statistics monitoring	Not supported.

Future enhancements to OpenVMS software will be based primarily on Structure Level 5; therefore, Structure Level 1 volumes might be further restricted in the future.

Appendix B. Tables of Time Differential Factors

The tables in this appendix show the time differential factors (TDFs) of various locations in the world. Each table contains a list of locations in a specific region. The information in the tables is believed to be accurate at the time of publication.

Note

Time zone rules are under control of each country, and are subject to change for political and other reasons. For up-to-date information, see the following web locations:

```
http://swissinfo.net/cgi/worldtime/
http://times.clari.net.au/index.htm
```

Table B.1, "TDFs for Europe" lists the time differential factors for Europe.

Table B.1. TDFs for Europe

Region	Standard Time TDF	Daylight Saving Time TDF
Great Britain, Ireland	0:00	+1:00
Western European Time	0:00	+1:00
Iceland	0:00	—
Middle European Time	+1:00	+2:00
Poland	+2:00	+3:00
Eastern European Time	+2:00	+3:00
Turkey	+2:00	+3:00

Table B.2, "TDFs for North America" lists the time differential factors for North America.

Table B.2. TDFs for North America

Region	Standard Time TDF	Daylight Saving Time TDF
U.S./Eastern Time	-5:00	-4:00
U.S./Central Time	-6:00	-5:00
U.S./Mountain Time	-7:00	-6:00
U.S./Pacific Time	-8:00	-7:00
U.S./Indiana (East)	-5:00	—
U.S./Alaska	-9:00	-8:00
U.S./Arizona	-7:00	—
U.S./Navajo	-7:00	-6:00
U.S./Michigan	-5:00	-4:00

Region	Standard Time TDF	Daylight Saving Time TDF
U.S./Aleutian Islands	-10:00	-9:00
U.S./Hawaii	-10:00	—
U.S./Samoa	-11:00	—
Canada/Newfoundland	-3:30	-2:30
Canada/Atlantic	-4:00	-3:00
Canada/Eastern	-5:00	-4:00
Canada/Central	-6:00	-5:00
Canada/East–Saskatchewan	-6:00	—
Canada/Mountain	-7:00	-6:00
Canada/Pacific	-8:00	-7:00
Canada/Yukon	-9:00	-8:00

Table B.3, "TDFs for Central and South America" lists the time differential factors for Central and South America.

Table B.3. TDFs for Central and South America

Region	Standard Time TDF	Daylight Saving Time TDF
Mexico/BajaNorte	-8:00	-7:00
Mexico/BajaSur	-7:00	—
Mexico/General	-6:00	—
Cuba	-5:00	-4:00
Jamaica	-5:00	-4:00
Brazil/East	-3:00	-2:00
Brazil/West	-4:00	-3:00
Brazil/Acre	-5:00	-4:00
Brazil/DeNoronha	-2:00	-1:00
Chile/Regional	-4:00	-3:00
Chile/Easter Island	-6:00	-5:00

Table B.4, "TDFs for Asia" lists the time differential factors for Asia.

Table B.4. TDFs for Asia

Region	Standard Time TDF	Daylight Saving Time TDF
PRC (Mainland China)	+8:00	+9:00
ROK (Korea)	+9:00	+10:00
Israel	+2:00	+3:00
Iran	+3:30	+4:30
Japan	+9:00	—

Region	Standard Time TDF	Daylight Saving Time TDF
Singapore	+8:00	—
Hong Kong	+8:00	—
ROC (Taiwan)	+8:00	—

Table B.5, "TDFs for the South Pacific" lists the time differential factors for the South Pacific.

Table B.5. TDFs for the South Pacific

Region	Standard Time TDF	Daylight Saving Time TDF
Australia/Tasmania	+10:00	+11:00
Australia/Queensland (standard time only)	+10:00	—
Australia/Queensland	+10:00	+11:00
Australia/North	+9:30	—
Australia/West	+8:00	—
Australia/South	+9:30	+10:30
Australia/Victoria	+10:00	+11:00
Australia/New South Wales	+10:00	+11:00
New Zealand	+12:00	+13:00

Table B.6, "TDFs for Antarctica" lists the time differential factors for Antarctica.

Table B.6. TDFs for Antarctica

Region	Standard Time TDF	Daylight Saving Time TDF
Antarctica	+0:00	—

Appendix C. VSI MIB Subagents Implemented on OpenVMS Alpha

The Extensible Simple Network Management Protocol (eSNMP) allows network managers to manage many different types of devices across all network and vendor boundaries through the use of databases called Management Information Bases (MIBs). Essentially, information is exchanged between **master agents** and **subagents**, which are devices such as routers and servers on the network being managed, and **managers**, which are the devices on the network through which the management is done.

This appendix describes the VSI Server MIB and the VSI Cluster MIB.

C.1. VSI Server MIB Subagents

The VSI Server MIB (DSM) consists of two extensions, or subagents:

Extension	Describes
System	A management interface to Alpha system information not defined by standard MIBs
Management	Instrumentation in the VSI extension agent, including the ability to detect and monitor thresholds on integer variables

The representation of the DSM within the standard Structure of Managed Information (SMI) framework is:

```
iso(1) org(3) dod(6) internet(1) private(4) enterprises(1) 36
```

OpenVMS Alpha Version 7.1-1H1 and later implements the DSM subagents on the AlphaServer 800, 1000, 4000, 4100, 8200, and 8400 systems. With the DSM subagents, customers can remotely determine and manage important information such as:

- Firmware revision numbers
- Base system descriptions
- Field Replaceable Unit (FRU) information and descriptions
- Processor and cache status
- Interface configurations
- Environmental conditions in the system enclosure that might be detrimental to the hardware

Use the following software to access the DSM subagents:

- The VSI ServerWORKS Manager Version 3.0 or any MIB browser that has access to the DSM definitions.
- VSI TCP/IP Services for OpenVMS Version 4.1 or later. The DSM subagents use the SNMP agent supplied with TCP/IP Services to communicate with SNMP clients.

The following sections describe the DSM subagents and explain how to set up your system to use them.

C.1.1. Overview of DSM Subagents

DSM subagents respond to SNMP requests for a DSM **object—** the data item that the network manager is concerned with, or a **trap —** information about a change of status. A subagent is responsible for reporting on and maintaining the data pertaining to these objects and traps.

The DSM system subagent implements the objects listed in *Table C.1, "DSM System Subagent Objects Implemented on OpenVMS Alpha"*. Each object corresponds to a group of base system and environmental information relevant to OpenVMS Alpha networking and can be accessed by a network manager through Server WORKS Manager.

Table C.1. DSM System Subagent Objects Implemented on OpenVMS Alpha

Object	Data Type	Access	Description
MIB Information Group			
svrSysMibMajorRev	Integer	Read only	The major revision number of this implementation of the svrSystem MIB. Currently 1.
svrSysMibMinorRev	Integer	Read only	The minor revision number of this implementation of the svrSystem MIB. Currently 0.
Base System Description Group			
svrSystemModel	DisplayString	Read only	System and model name. For example, AlphaServer 2100.
svrSystemDescr	DisplayString	Read only	General text description of system type.
svrSystemBoardFruIndex	Integer	Read only	The index of the Field Replaceable Unit (FRU) in the FRU table describing the serial number and other asset information of the board. If unknown, 0.
svrSystemBootedOS	Integer	Read only	The current booted operating system.
svrSystemShutdownReason	DisplayString	Read only	The possible reason for the system shutdown.
svrFirmwareIndex	Integer	Read only	An index value unique to the local system.
svrFirmwareDescr	DisplayString	Read only	Descriptive text for items such as the SRM console, ARC console, and system BIOS.
svrFirmwareRev	DisplayString	Read only	A version number, often of the form V *x.y* or V *x.y-z*.
svrFwSymbolName	DisplayString	Read only	The symbol name as visible on the console.
svrFwSymbolValue	Octet string	Read only	The symbol value. Null if none or unknown.
System Processor Group			

Object	Data Type	Access	Description
svrCpuIndex	Integer	Read only	An index value for the CPU entry that is unique to the local system.
svrCpuManufacturer	DisplayString	Read only	The manufacturer of the processor.
svrCpuRevision	DisplayString	Read only	Version information in processor-specific format.
svrCpuFruIndex	Integer	Read only	The index of the FRU entry in the FRU table that describes the asset information of the component containing the processor. If unknown, 0.
svrCpuCacheIndex	Integer	Read only	The local index value.
svrCpuCacheLevel	Integer	Read only	Level 1, level 2, level 3 cache, other, or unknown.
svrCpuCacheType	Integer	Read only	Type of cache: internal, external, internal instruction, or internal data.
svrCpuCacheSize	Kbytes	Read only	Cache size in Kbytes.
svrCpuCacheSpeed	Integer	Read only	Cache speed in nanoseconds. If unknown, 0.
svrCpuCacheStatus	Integer	Read only	Current status of the cache: enabled, disabled, other, or unknown.
Memory Configuration Group			
svrPhysicalMemorySize	Kbytes	Read only	Total amount of physical memory as seen by the operating system.
svrPhysicalMemoryFree	Kbytes	Read only	Amount of free physical memory.
svrMemIndex	Integer	Read only	Unique index for this entry.
svrMemSize	Kbytes	Read only	Length of memory range.
svrMemFruIndex	Integer	Read only	Index of the FRU entry in the FRU table on which the memory resides. If unknown, 0.
svrBusIndex	Integer	Read only	An index value that is unique to the local system.
svrBusType	BusTypes	Read only	Bus type.
svrLogicalSlotNumber	Integer	Read only	Unique logicals lot number on a given bus.
svrLogicalSlotDescr	DisplayString	Read only	Device description derived from ID or as set by the management station.
svrLogicalSlotRevision	DisplayString	Read only	Vendor-supplied major and minor revision of device in the slot.
Physical Configuration Group			
svrFruIndex	Integer	Read only	An index value that is unique to the system.
svrFruType	Integer	Read only	General category of FRU type.

Object	Data Type	Access	Description
svrFruDescr	DisplayString	Read only	Detailed description of the FRU type, if known.
svrFruVendor	DisplayString	Read only	Manufacturer's name or ID.
svrFruPartNumber	DisplayString	Read only	Order number for this unit.
svrFruRevision	DisplayString	Read only	Version number of the unit. If an illustration is available, it appears as “Artwork: XXX” following the FRU version number.
svrFruFirmwareRevision	DisplayString	Read only	Revision of the firmware, if applicable. Otherwise, null.
svrFruSerialNumber	DisplayString	Read only	Unit's serial number.
Environment Group: Thermal			
svrThermalSensorCount	Integer	Read only	Number of thermal sensors present and readable in the system.
svrThSensorIndex	Integer	Read only	An index value unique to the local system.
svrThSensorReading	Integer	Read only	Current value read by the sensor in units as described by the svrThSensorReadingUnits object.
svrThSensorReadingUnits	ThermUnits	Read only	Value of sensor in degrees Fahrenheit, Celsius, or relative value. If not available, value will be unknown.
svrThSensorStatus	Integer	Read only	The sensor's status value.
Environment Group: Cooling			
svrFanCount	Integer	Read only	The number of fans whose states are detectable.
svrFanIndex	Integer	Read only	An index value unique to the local system.
svrFanStatus	Integer	Read only	Current fan status.
Environment Group: Power Supply			
svrPowerSupplyCount	Integer	Read only	Number of detectable power supplies reflected as entries in the svrPowerSupplyTable object.
svrPowerSupplyIndex	Integer	Read only	An index value unique to the local system.
svrPowerSupplyStatus	Integer	Read only	Current state of the power supply.

The DSM Management subagent implements the objects and traps listed in Tables *Table C.2, "DSM Management Subagent Objects Implemented on OpenVMS Alpha"* and *Table C.3, "DSM Management Subagent Traps Implemented on OpenVMS Alpha"*, respectively.

Each object or trap corresponds to a group of management areas relevant to OpenVMS Alpha networking and can be accessed by a network manager through ServerWORKS Manager.

Table C.2. DSM Management Subagent Objects Implemented on OpenVMS Alpha

Object	Data Type	Access	Description
MIB Information Group			
svrMgtMibMajorRev	Integer	Read only	The major revision number of this implementation of the svrMgtMIB. Currently 1.
svrMgtMibMinorRev	Integer	Read only	The minor revision number of this implementation of the svrMgtMIB. Currently 0.
Alarms Group			
svrAlarmNextThrIndex	Integer	Read only	The next available index for creating a svrThrEntry object. If the value is -1, the maximum number of thresholds has been reached. A threshold record cannot be created until you delete the current threshold record.
svrAlarmEnableTraps	Boolean	Read/write	If true, a trap is sent for each triggered alarm.
svrThresholdTable	Sequence of SvrThresholdEntry	Not accessible	The threshold table that describes conditions for setting and resetting alarms. The agent checks this table for exceptions. You can set alarms on absolute values (such as the current integer value of the sampled variable) or on delta values (such as the difference between the current or last value). Alarms can be `Greater Than` exception alarms, `Less Than` exception alarms, `Equal To` alarms, and so on. (See the svrThrAlarmType object description.) Hysteresis (the tendency of certain binary devices to show different threshold values when changing from 0 to 1 than when changing from 1 to 0) is introduced by providing thresholds both for setting and resetting of the alarm state, thereby limiting the number of traps that are sent on alarm triggering. You can create alarms to persist across agent reboots; however, this is not recommended for dynamic table variables. The triggering of an alarm changes a state variable in the conceptual row and

Object	Data Type	Access	Description
			can also trigger the sending of a trap, or the local logging of an event.
svrThresholdEntry	SvrThresholdEntry	Not accessible	A threshold alarm set on an integer variable. An alarm entry is created by the management console using the current value of svrAlarmNextThrIndex to name the instances of the row variables, setting the svrThrStatus to under Creation. When you create a threshold entry for the first time, issue a set request on svrThrStatus. You can set the remaining row variables in the same operation or in subsequent operations. Those not set retain their default values as described. You must set variable values for the following objects in the Alarms group before you enable the alarm: ● svrThrStatus (set to under Creation) ● svrThrVariableName through svrThrSeverity (set appropriately; seethe object descriptions)
svrThrIndex	Integer	Read only	An index value unique to the local system. On creation, set to the value of svrAlarmNextThrIndex.
svrThrStatus	Integer	Read/write	Describes the status of the row. When the row is created with the initial set, you must set svrThrStatus to under Creation. When the management console has completed the row setup, it sets this variable to row Enabled. Variables in the row can only be written if svrThrStatus is in the initial under Creation state or has been set to row Disabled. To delete the row, set the status to row Invalid. Be aware that errors in variable polling and threshold checking that cannot be corrected cause a row status change to row Error. Once the status is set to row Error by the agent, the agent does not reset the status. Instead, the management console must reset the status based on information returned

Object	Data Type	Access	Description
			with svrThrErrorValue or for other reasons.
svrThrVariableName	Object identifier	Read/write	The object identifier (OID) of an integer variable to be tested against the threshold. At row creation, the variable equals the value 0.0 and must be set to the OID of an integer variable before enabling the alarm.
svrThrValueType	Integer	Read/write	Absolute or delta value. The default on row creation is absolute Value. The delta Value is calculated by taking the current value and subtracting the svtThrLastValue value.
svrThrAlarmType	Integer	Read/write	An alarm that signals a threshold whose value is `Greater Than`, `Greater Than or Equal To`, `Equal To`, `Less Than or Equal To`, or `Less Than`. The default value on row creation is `GreaterThan`. `Greater Than` or `Greater Than or Equal To` thresholds for absolute values occur when the sample value equals or exceeds the svrThrThresholdValue and svrThrAlarmState was reset. This condition causes svrThrAlarmState to be set and, if svrAlarmEnableTraps is true, a svrThrExceptTrap is sent. SvrThrAlarmState is reset when the sample value falls below or equals svrThrResetValue. `Less Than` or `Less Than or Equal To`thresholds for absolute values occur when the sample value falls below or equals the svrThrThresholdValue, and svrThrAlarmState was reset. This condition causes the svrThrAlarmState to be set and, if svrAlarmEnableTraps is true, a svrThrExceptTrap is sent. SvrThrAlarmState is reset when the sample value exceeds or equals svrThrResetValue. `Equal To` thresholds for absolute values occur when the sample value equals svrThrThresholdValue and svrThrAlarmState was reset. This

Object	Data Type	Access	Description
			condition causes the svrThrAlarmState to be set and, if svrAlarmEnableTraps is true, a svrThrExceptTrap is sent. SvrThrAlarmState is reset when the sample value does not equal svrThrResetValue. The same conditions apply for delta values as for absolute values except the difference between the sample value and the svrThrLastValue is used for comparison with both the svrThrThresholdValue and the svrThrResetValue. Note that it is possible to have negative delta values since the difference is computed as the current value minus the svrThrLastValue.
svrThrSampleInterval	Integer	Read/write	The interval (in seconds) between polls to check for threshold exceptions. The default value on row creation is 30 seconds. Minimum value: 1.

svrThrPersistent	Boolean	Read/write	If true, the threshold persists across agent restarts. Default on row creation: false.
			
			By default, the files used to store persistent data are SYS$SYSTEM: TCPIP$MGT_THRESHOLDS.DAT and SYS$SYSTEM: TCPIP $MGT_THRESHOLDS.BAK. To move the files off the system disk or rename them, the system manager can define the logical names TCPIP $MGT_PERSISTENCE_DAT and TCPIP $MGT_PERSISTENCE_BAK in the SYS$MANAGER:SYLOGICALS.COM file as appropriate. For example, to point to files in a different location, add the following definitions to SYS $MANAGER:SYLOGICALS.COM. (The examples are formatted to fit the table column.)
			
			`$ DEFINE/SYS -`
			`_$ TCPIP$MGT_PERSISTENCE_DAT -`
			`_$ DISK2: [SNMP.MIB]PERSIST.DAT;`
			`$ DEFINE/SYS -`

Object	Data Type	Access	Description
			_$ **TCPIP$MGT_PERSISTENCE_BAK -** _$ **DISK2:[SNMP.MIB]PERSIST.BAK;**
svrThrThresholdValue	Integer	Read/write	The threshold value that is compared to the current or delta value. Default on row creation: 0.
svrThrResetValue	Integer	Read/write	The value used to reset the threshold on all svrThrAlarmTypes objects except those that are `Equal To`. Default on row creation: 0.
svrThrLastValue	Integer	Read only	The previous sample needed to evaluate if alarm should be triggered or to evaluate delta values for threshold checking.
svrThrAlarmState	Integer	Read only	Indicates whether the alarm is currently set or reset. Used by polling management applications to determine if a threshold exception state has been detected based on the alarm definition. Has an initial value of reset when the alarm is enabled or the agent is restarted. The value is reset if svrThrStatus changes to rowDisabled or rowInvalid. For guidelines on state changes, see the description for svrThrAlarmType.
svrThrLogEvent	Boolean	Read/write	If true, logs data to the subagent process log file; for example, to [TCPIP $SNMP]TCPIP$SVRMGT_MIB.LOG. (See *Section C.1.2, "Setting Up the System to Use the DSM Subagents"*.) Default value: false.
svrThrDescr	DisplayString	Read/write	Describes the type of threshold. Set by the management console, not by the agent.
svrThrErrorValue	SnmpErrors	Read only	The SNMP-defined error status that caused the svrThrStatus value to become equal to rowError. Valid only at that time.
svrThrComparisonName	Object identifier	Read/write	An object identifier (OID) to a descriptor attribute used with the svrThrPersistent value to verify that the svrThrVariableName instance is correct. Optional. Default:0.0. On agent restarts, the value is retrieved and compared to the svrThrComparisonValue. If not

Object	Data Type	Access	Description
			equal, the OID instancing for svrThrVariableName might be incorrect. If this situation occurs, svrThrStatus is set to rowError and svrThrErrorValue to bad Value.
svrThrComparisonValue	DisplayString	Read/write	Date value of svrThrComparisonName. Optional. Used when svrThrPersistent is set. The value is compared to the current value on agent restarts. Default: null.
svrThrSeverity	Severity	Read/write	Indicates the severity of the threshold. Default on row creation: informational.

Table C.3. DSM Management Subagent Traps Implemented on OpenVMS Alpha

Trap	Variable	Description
Local Server Control Group		
svrThrHighExceptTrap	svrThrVariableName svrThrValueType svrThrThresholdValue svrThrLastValue svrThrDescr	A high severity trap. The value that caused the alarm to occur is returned in svrThrLastValue.
svrThrMediumExceptTrap	svrThrVariableName svrThrValueType svrThrThresholdValue svrThrLastValue svrThrDescr	A medium severity trap. The value that caused the alarm to occur is returned in svrThrLastValue.
svrThrLowExceptTrap	svrThrVariableName svrThrValueType svrThrThresholdValue svrThrLastValue svrThrDescr	A low severity trap. The value that caused the alarm to occur is returned in svrThrLastValue.
svrThrInformationalExceptTrap	svrThrVariableName svrThrValueType svrThrThresholdValue svrThrLastValue svrThrDescr	An Informational severity trap. The value that caused the alarm to occur is returned in svrThrLastValue.

C.1.2. Setting Up the System to Use the DSM Subagents

To configure SNMP on the system and enable the master agent to accept SET commands from SNMP clients, issue the following TCP/IP Services management command from the TCPIP> prompt. This operation requires SYSPRV or BYPASS privileges.

```
TCPIP> SET CONFIGURATION SNMP /FLAGS=SETS
```

To enable or disable the type of access to your local MIB data, use the following commands, qualifiers, and options:

```
TCPIP> SET CONFIGURATION SNMP /[NO]COMMUNITY="name" -
_TCPIP> /[NO]ADDRESS=host address -
_TCPIP> /TYPE=[NO]READ, [NO]TRAP, [NO]WRITE
```

For example, the following command configures SNMP, specifies the community name and address, specifies that the agent can accept SET commands from members of the community, and enables the master agent to send trap messages to members of the community. (Note that READ access is assumed when specifying TRAP or WRITE.)

```
TCPIP> SET CONFIGURATION SNMP /COMMUNITY="public" /ADDRESS=128.45.2.8 -
_TCPIP> /TYPE=TRAP, WRITE
```

For your convenience, the following files have example entries to start, run, and shut down the DSM subagents.

- [TCPIP$SNMP]TCPIP$EXTENSION_MIB_STARTUP.COM. (This file is called by TCPIP $SNMP_STARTUP.COM.)
- [TCPIP$SNMP]TCPIP$EXTENSION_MIB_RUN.COM. (This file is called by TCPIP $SNMP_RUN.COM.)
- [TCPIP$SNMP]TCPIP$EXTENSION_MIB_SHUTDOWN.COM. (This file is called by TCPIP $SNMP_SHUTDOWN.COM.)

Search the files for SVRSYSTEM_MIB entries and edit them as needed. You must also remove the GOTO entries, which cause the command procedures to simply exit.

C.2. VSI Cluster MIB Subagents

The VSI Cluster MIB (DCM) is a private VSI management information base that delivers management information about OpenVMS Cluster systems.

The DCM consists of two extensions, or subagents:

Extension	Describes
System	A management interface to cluster system information not defined by standard MIBs
Management	Instrumentation in the VSI extension agent, including the ability to detect and monitor thresholds on integer variables

The representation of the DCM within the standard Structure of Managed Information (SMI) framework is:

```
iso(1) org(3) dod(6) internet(1) private(4) enterprises(1) 36
```

OpenVMS Alpha Version 7.2 implements the DCM subagents. With the DCM subagents, you can remotely determine status information for an OpenVMS Cluster system, including the following:

- Cluster software version
- Status of the cluster software: installed, running, failed, and so forth
- Type of cluster that is running
- State change to reflect when a cluster member is added or deleted

Use the following software to access the DCM subagents:

- The Server WORKS Manager Version 3.0 or any MIB browser that has access to the DCM definitions.
- VSI TCP/IP Services for OpenVMS Version 4.1 or later. The DCM subagents use the SNMP agent supplied with TCP/IP Services to communicate with SNMP clients.

The following sections describe the DCM subagents and explain how to set up your system to use them.

C.2.1. Overview of DCM Subagents

DCM subagents respond to SNMP requests for a DCM **object—** the data item that the network manager is concerned with, or a **trap —** information about a change of status. A subagent is responsible for reporting on and maintaining the data pertaining to these objects and traps.

The DCM subagents implement the objects listed in *Table C.4, "DCM Subagent Objects Implemented on OpenVMS"*. Each object returns information relevant to an OpenVMS Cluster system and can be accessed by a network manager through Server WORKS Manager.

Table C.4. DCM Subagent Objects Implemented on OpenVMS

Object	Data Type	Access	Description
Cluster Information			
svrCluSoftwareVendor	DisplayString	Read only	Cluster software vendor name.
svrCluSoftwareVersion	DisplayString	Read only	Cluster software version. This is the OpenVMS version string.
svrCluSoftwareStatus	ClusterStatus	Read only	The status of the cluster software. Possible values are running and not running.
svrCluClusterType	ClusterType	Read only	The type of cluster that is running. The current value is OpenVMS.
svrCluExtensionOID	Object Identifier	Read only	The authoritative identification for the MIB. If no such identifier exists, the value {0.0} is returned.
svrCluThisMember	Integer	Read only	Index into the member table (svrCluMemberTable) that corresponds to this node.
SMNP Traps			

Object	Data Type	Access	Description
svrCluMemberAdded	Trap Packet	Read only	Generated when a cluster member is added.
svrCluMemberDeleted	Trap Packet	Read only	Generated when a cluster member is deleted.
Node-Specific Information			
svrCluMemberIndex	Integer	Read only	A unique index for the entry. Values of svrCluMemberIndex must remain constant at least between reboots of the network management system on the managed node. Where possible, this value should reflect the system's native member identifier.
svrCluMemberName	DisplayString	Read only	The SCS node name of this cluster member. A zero-length value means the member's node name is unknown. This name may not necessarily resolve to an address.
svrCluMemberComment	DisplayString	Read only	This is the hardware name of the node, as returned by the $GETSYI system service.
svrCluMemberStatus	MemberStatus	Read only	Status of this member. Possible values are normal and removed.
svrCluMemberAddressIndex	Integer	Read only	The index for this address.
svrCluMemberAddress	IpAddress	Read only	An IP address of this cluster member. This address may not be reachable from nodes that are not configured into the cluster.

C.2.2. Setting Up the System to Use the DCM Subagents

For your convenience, the following files have example entries to start, run, and shut down the DCM subagents.

- [TCPIP$SNMP]TCPIP$EXTENSION_MIB_STARTUP.COM. (This file is called by TCPIP $SNMP_STARTUP.COM.)
- [TCPIP$SNMP]TCPIP$EXTENSION_MIB_RUN.COM. (This file is called by TCPIP $SNMP_RUN.COM.)
- [TCPIP$SNMP]TCPIP$EXTENSION_MIB_SHUTDOWN.COM. (This file is called by TCPIP $SNMP_SHUTDOWN.COM.)

Search the files for SVRCLUSTER_MIB entries and edit them as needed. You must also remove the `GOTO` entries, which cause the command procedures to simply exit.