

VMS Version 5.2 New Features Manual

Order Number: AA-LA97B-TE

June 1989

This manual describes the new features of Version 5.2 of the VMS operating system.

Revision/Update Information: This manual supersedes the *VMS Version 5.0 New Features Manual*.

Software Version: VMS Version 5.2

**digital equipment corporation
maynard, massachusetts**

June 1989

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital Equipment Corporation or its affiliated companies.


© Digital Equipment Corporation 1989.

All Rights Reserved.

Printed in U.S.A.

The postpaid Reader's Comments forms at the end of this document request your critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

CDA	MASSBUS	VAX RMS
DDIF	PrintServer 40	VAXstation
DEC	Q-bus	VMS
DECnet	ReGIS	VT
DECUS	ULTRIX	XUI
DECwindows	UNIBUS	
DIGITAL	VAX	
LN03	VAXcluster	

The following is a third-party trademark:

PostScript is a registered trademark of Adobe Systems, Inc.

ZK5215

Production Note

This book was produced with the VAX DOCUMENT electronic publishing system, a software tool developed and sold by Digital. In this system, writers use an ASCII text editor to create source files containing text and English-like code; this code labels the structural elements of the document, such as chapters, paragraphs, and tables. The VAX DOCUMENT software, which runs on the VMS operating system, interprets the code to format the text, generate a table of contents and index, and paginate the entire document. Writers can print the document on the terminal or line printer, or they can use Digital-supported devices, such as the LN03 laser printer and PostScript printers (PrintServer 40 or LN03R ScriptPrinter), to produce a typeset-quality copy containing integrated graphics.

Contents

Preface

xi

Chapter 1 General User Features: DCL Commands

1.1	ANALYZE/AUDIT Command	1-2
1.2	SET DISPLAY Command	1-3
1.3	SHOW DISPLAY Command	1-8
1.4	DISMOUNT Qualifier	1-11
1.5	LOGIN Procedure Qualifier	1-11
1.6	SET AUDIT Command	1-11
1.7	SET FILE Qualifier: /[NO]SEMANTICS	1-20
1.8	SHOW AUDIT Command	1-21
1.9	SHOW SYSTEM Qualifiers	1-23
1.10	SHOW USERS Command	1-23
1.11	Lexical Function: F\$CONTEXT	1-24
1.12	Lexical Function Item Codes	1-29

Chapter 2 System Management Features

2.1	Security Auditing	2-1
2.1.1	Audit Server Database	2-2
2.1.2	Audit Server Process	2-3
2.1.3	VMS Audit Analysis Utility	2-4
2.2	SYSGEN Utility	2-5
2.2.1	DEINSTALL Command Description	2-5
2.2.2	ERLBUFFERPAGES Parameter	2-6
2.3	SYSMAN Utility	2-6

2.3.1	ALF Support Added to SYSMAN	2-6
2.3.2	Change in Privileges Granted on Remote Nodes	2-8
2.3.3	CONFIGURATION SHOW CLUSTER Command-Enhanced Display	2-9
2.3.4	LICENSE Commands	2-9
2.3.5	SET ENVIRONMENT/NODE Command	2-10
2.4	License Management Facility	2-11
2.5	NETCONFIG.COM Security Enhancements	2-11
2.5.1	Default Access Options	2-12
2.5.2	Security Benefits	2-13
2.5.3	Questions Posed by NETCONFIG.COM	2-13
2.6	New NETCONFIG_UPDATE.COM for Existing Networks	2-14
2.6.1	Benefits of NETCONFIG_UPDATE.COM	2-14
2.6.2	Using NETCONFIG_UPDATE.COM in a VAXcluster	2-14
2.7	Disk and Tape Features	2-14
2.8	Backup Utility	2-15
2.8.1	Performance Enhancements	2-15
2.8.2	Cyclic Redundancy Checking Emulation Improvements	2-18
2.8.3	Pressing Ctrl/T to Obtain Information About BACKUP Operations	2-19
2.9	VMS DECwindows System Tailoring	2-19

Chapter 3 Programming Features

3.1	Debugger	3-1
3.2	System Services	3-2
3.3	VMSINSTAL New Features	3-3
3.3.1	Check_Network Callback	3-3
3.3.2	CHECK_PRODUCT_VERSION Callback	3-4
3.3.3	COMPARE_IMAGE Callback	3-5
3.3.4	GET_IMAGE_ID Callback	3-6
3.3.5	UPDATE_IDENTIFIER Callback	3-7
3.3.6	SET Callback—New ASK_CASE Option	3-7
3.3.7	SET Callback—New POSTINSTAL Option	3-8
3.3.8	CHECK_NET_UTILIZATION Callback—New Parameters	3-8

3.3.9	CREATE_DIRECTORY Callback—New Option	3-9
3.3.10	ASK Callback—New Options	3-10
3.3.11	VMSINSTAL Command Line—New and Changed Options	3-10

Chapter 4 System Services Features

4.1	Modifications to \$SETUAI and \$GETUAI	4-1
4.1.1	New Item Codes for \$SETUAI and \$GETUAI ..	4-2
4.1.2	New Authorization Flags for \$SETUAI and \$GETUAI	4-3
4.2	Modifications to \$MOUNT	4-4
4.3	Modifications to \$DISMOUNT	4-4
4.4	Modifications to Existing System Services for Clusterwide Process Accessibility	4-5
4.5	Process Information Services	4-6
4.6	Overview of \$GETJPI and \$GETJPI with \$PROCESS_SCAN	4-6
4.6.1	Using the Process ID to Obtain Information ...	4-7
4.6.2	Using the Process Name to Obtain Information	4-8
4.6.3	Modifications to \$GETJPI	4-9
4.7	Using \$GETJPI Alone	4-9
4.7.1	Requesting Information About a Single Process	4-10
4.7.2	Requesting Information About All Processes on the Local System	4-12
4.8	Using \$GETJPI with \$PROCESS_SCAN	4-14
4.8.1	Using the \$PROCESS_SCAN Item List and Item-Specific Flags	4-14
4.8.2	Requesting Information About Processes That Match One Criterion	4-15
4.8.3	Requesting Information About Processes That Match Multiple Values for One Criterion	4-18
4.8.4	Requesting Information About Processes That Match Multiple Criteria	4-19
4.8.5	Specifying a Node as Selection Criterion	4-20
4.8.6	Scanning All Nodes on the Cluster for Processes	4-21

4.8.7	Scanning Specific Nodes on the Cluster for Processes	4-21
4.8.8	Conducting Multiple Simultaneous Searches with \$PROCESS_SCAN	4-22
4.9	Programming Considerations for GETJPI\$	4-23
4.9.1	Using Item Lists Correctly	4-23
4.9.2	Improving Performance by Using Buffered \$GETJPI Operations	4-23
4.9.3	Meeting Remote \$GETJPI Quota Requirements	4-25
4.9.4	Using \$GETJPI Control Flags	4-26
4.10	\$PROCESS_SCAN and \$DEVICE_SCAN System Services	4-31
	\$DEVICE_SCAN	4-32
	\$PROCESS_SCAN	4-36

Chapter 5 System Error Messages

5.1	New Messages	5-1
5.2	Changes to Existing Messages	5-22

Appendix A Version 5.1 Features

A.1	Support for Compound Documents	A-1
A.1.1	VMS Commands and Utilities	A-2
A.1.2	DDIF Support in a Heterogeneous Environment	A-6
A.1.3	VMS RMS Interface Changes	A-8
	\$XABITM	A-10
A.1.4	Distributed File System Support for DDIF Tagged Files	A-16
A.1.5	VMS RMS Errors	A-16
A.2	EXCHANGE/NETWORK Command	A-17
	EXCHANGE/NETWORK	A-18

Index

Examples

4-1	Using \$GETJPI to Obtain Information About the Calling Process	4-10
4-2	Using \$GETJPI and the Process Name to Obtain Information About a Process	4-12
4-3	Using \$GETJPI to Request Information About All Processes on the Local System	4-13
4-4	Using \$GETJPI and \$PROCESS_SCAN to Select Process Information by User Name	4-16
4-5	Using \$GETJPI and \$PROCESS_SCAN with Multiple Values for One Criterion	4-19
4-6	Selecting Processes That Match Multiple Criteria	4-20
4-7	Searching the Cluster for Process Information	4-21
4-8	Searching for Process Information on Specific Nodes in the Cluster	4-22
4-9	Using a \$GETJPI Buffer to Improve Performance	4-24
4-10	Using \$GETJPI Control Flags to Avoid Swapping a Process into the Balance Set	4-28
A-1	Tagging a File	A-11
A-2	Accessing a Tagged File	A-14

Figures

1-1	Running Remote and Local Applications	1-6
-----	---	-----

Tables

2-1	New Security Auditing Features	2-2
2-2	UAF Process Quotas for the BACKUP Account	2-16
2-3	Suggested Values for UAF Process Quotas	2-17
4-1	Process Identification	4-8
A-1	Tag Support Item Codes	A-8

Preface

Intended Audience

This book is intended for general users, system managers, and programmers who use the VMS operating system.

Document Structure

This book is organized as follows:

- Chapter 1 describes new and changed DCL commands.
- Chapter 2 describes new system management features.
- Chapter 3 describes new programming features.
- Chapter 4 describes new and changed system services.
- Chapter 5 describes new system messages.
- Appendix A describes features that were new for Version 5.1 but have not been incorporated into the documentation set.

Associated Documents

The following books provide additional information about Version 5.2:

- *VMS Debugger Manual*
- *VMS Audit Analysis Utility Manual*
- *VMS License Management Utility Manual*
- *VMS Version 5.2 Release Notes*

Conventions

The following conventions are used in this manual:

mouse	The term <i>mouse</i> is used to refer to any pointing device, such as a mouse, a puck, or a stylus.
MB1, MB2, MB3	MB1 indicates the left mouse button, MB2 indicates the middle mouse button, and MB3 indicates the right mouse button. (The buttons can be redefined by the user.)
PB1, PB2, PB3, PB4	PB1, PB2, PB3, and PB4 indicate buttons on the puck.
SB1, SB2	SB1 and SB2 indicate buttons on the stylus.
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, then press and release another key or a pointing device button.
<code>Return</code>	A key name is shown enclosed to indicate that you press a key on the keyboard.
...	In examples, a horizontal ellipsis indicates one of the following possibilities: <ul style="list-style-type: none">• 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 format descriptions, parentheses indicate that, if you choose more than one option, you must enclose the choices in parentheses.
[]	In format descriptions, brackets indicate that whatever is enclosed is optional; you can select none, one, or all of the choices.
{}	In format descriptions, braces surround a required choice of options; you must choose one of the options listed.
red ink	Red ink indicates information that you must enter from the keyboard or a screen object that you must choose or click on. For online versions, user input is shown in bold .
boldface text	Boldface text represents the introduction of a new term or the name of an argument, an attribute, or a reason.

<i>italic text</i>	Italic text represents information that can vary in system messages (for example, Internal error <i>number</i>).
UPPERCASE TEXT	Uppercase letters indicate that you must enter a command (for example, enter OPEN/READ).
UPPERCASE TEXT	Uppercase letters indicate the name of a routine, the name of a file, the name of a file protection code, or the abbreviation for a system privilege.
-	Hyphens in coding examples indicate that additional arguments to the request are provided on the line that follows.
numbers	Unless otherwise noted, all numbers in the text are assumed to be decimal. Nondecimal radices—binary, octal, or hexadecimal—are explicitly indicated.

Chapter 1

General User Features: DCL Commands

VMS Version 5.2 includes the following new DCL features:

Feature	Function
New Commands	
ANALYZE/AUDIT	Extracts selective records from the system security audit journal.
SET DISPLAY	Allows creation and modification of workstation display devices.
SHOW DISPLAY	Displays the characteristics of a display device.
New Qualifiers	
DISMOUNT qualifier:	
/OVERRIDE=CHECKS	Marks a Files-11 volume for dismounting even if files are open on the volume.
LOGIN Procedure qualifier:	
/NEW_PASSWORD	Changes your password when you log in.
SET AUDIT qualifiers	Modifies the characteristics of the audit server; enables long-term journaling (<i>archiving</i>) of security audit messages; modifies resource monitoring characteristics.
SET FILE qualifiers:	
/SEMANTICS	Creates or changes a semantics tag.
/NOSEMANTICS	Removes a semantics tag.
SHOW AUDIT qualifiers	Displays security auditing characteristics in effect on the system.
SHOW SYSTEM qualifiers:	
/CLUSTER	Displays all processes on all nodes in a VAXcluster.
/NODE[=(name,...)]	Displays all processes on the specified node or nodes in a VAXcluster. The qualifier /NODE without a value displays all processes on the local node.

1-2 General User Features: DCL Commands

Feature	Function
New Qualifiers	
SHOW USERS qualifiers:	
/BATCH	Displays all batch users in the system.
/CLUSTER	Displays all users on all nodes in a VAXcluster.
/FULL	Displays complete information, rather than the default summary, about each process.
/INTERACTIVE	Displays all interactive users in the system.
/NETWORK	Displays all network users in the system.
/NODE[=(name,...)]	Displays all users on the specified node or nodes in a VAXcluster. The qualifier /NODE without a value displays all users on the local node.
/SUBPROCESS	Displays all subprocess users in the system.

New Lexical Function

F\$CONTEXT	Formats data for use with the SYS\$PROCESS_SCAN service. Sets up selection criteria for use with the F\$PID function.
------------	---

New Lexical Function Item Codes

F\$FILE_ATTRIBUTES item codes	Return information about the following: RMS journaling, file system options, stored semantics.
----------------------------------	--

The following sections describe these features in detail.

1.1 ANALYZE/AUDIT Command

The new clusterwide Audit Analysis Utility, invoked with the DCL command ANALYZE/AUDIT, improves the site security administrator's ability to review the security events logged to the system. Prior to VMS Version 5.2, all security event messages enabled with the DCL command SET AUDIT were copied to the operator log file (OPERATOR.LOG in the SYS\$MANAGER directory). VMS provided the DCL command procedure SECAUDIT.COM to extract security messages from the operator log file. For Version 5.2, SECAUDIT.COM is superseded by the Audit Analysis Utility.

Security event messages can be extracted from the system security audit journal and displayed on a terminal (or output to a file) using combinations of ANALYZE/AUDIT selection qualifiers. The Audit Analysis Utility also features an interactive, screen-oriented command mode, which can be invoked by entering Ctrl/C during the display. Once you enter interactive command mode, you can specify a record number for viewing, specify a different set of selection and

exclusion criteria for viewing audit records, obtain online help, and move forward or backwards through the security audit log file by any number of records.

For more information about the Audit Analysis Utility, see the *VMS Audit Analysis Utility Manual*.

1.2 SET DISPLAY Command

Use the SET DISPLAY command to direct the output of a VMS DECwindows application. Output can be directed from any VAX computer running a DECwindows application (including workstations) to any DECwindows workstation.

Both source and destination nodes must be part of the same network.

Specify SET DISPLAY in the following format:

SET DISPLAY [*display-device*]

Parameter

display-device

Specifies a logical name for the workstation display you are creating or modifying. If you are directing application output to multiple workstation displays, you can use different logical names to point to each display. If you do not specify a display-device string, the logical name DECW\$DISPLAY is used. This means that by default, application output will be displayed on the workstation display referred to by DECW\$DISPLAY.

By entering the command SHOW DISPLAY, you can see the workstation node where applications will be displayed by default. If you specified your own logical display device in the SET DISPLAY/CREATE command, include that logical name in the SHOW DISPLAY command.

Qualifiers

/CREATE

Creates the workstation display device (WSAn:) on which a DECwindows application is displayed. You must specify the /CREATE qualifier the first time you use the SET DISPLAY command, but you do not need to respecify it if you continue to redirect output from applications to other workstations with subsequent SET DISPLAY commands.

When /CREATE is specified without /NODE, the workstation device defaults to the current node.

/[NO]PERMANENT

Cancels the redirected display by deassigning the logical name DECW\$DISPLAY. If you specified a logical name as the display-device parameter with the SET DISPLAY/CREATE command, entering the SET DISPLAY/NOPERMANENT

1-4 General User Features: DCL Commands

SET DISPLAY Command

display-device command cancels the redirected display by deassigning the logical name you specified.

The Session Manager defines DECW\$DISPLAY in your job logical name table when you open a terminal (DECTerm) window. When you direct the application output display to another workstation with the SET DISPLAY/CREATE command, an additional DECW\$DISPLAY logical name is defined in your process logical name table. This definition supersedes the definition in the job logical name table. Output from applications run from the process in which you executed the SET DISPLAY/CREATE command will be displayed on the workstation device referred to by the definition of DECW\$DISPLAY in the process logical name table. Enter the SHOW DISPLAY command to see where this application will be displayed. To see whether multiple definitions for DECW\$DISPLAY exist, enter the command SHOW LOGICAL DECW\$DISPLAY.

If DECW\$DISPLAY is still defined (for example, in the job logical name table) after you specify the /NOPERMANENT qualifier, any DECwindows applications run from this process will be displayed on the workstation device to which output is now directed. Enter the SHOW DISPLAY command if you are unsure of the node to which DECW\$DISPLAY refers.

Use caution when entering the SET DISPLAY/NOPERMANENT command. If you modify or delete the definition of DECW\$DISPLAY from the job logical name table, you will be unable to start another session. Be careful not to specify the /NOPERMANENT qualifier without having first directed the display with the SET DISPLAY/CREATE command.

You cannot specify /NOPERMANENT and /CREATE on the same command line.

/NODE=workstation_display

Defines the workstation on which you want to display DECwindows applications. The node name you provide cannot be a cluster alias (a name that represents multiple nodes configured in a VAXcluster), but must instead identify an actual node.

You must create a workstation display device with the /CREATE qualifier before you can direct the output from applications to other workstations. Do not enter the SET DISPLAY/NODE=workstation_display command without having previously specified the /CREATE qualifier.

Make sure that you are authorized to display applications on the workstation node you specify. See the *VMS DECwindows User's Guide* for more information about using the DECwindows Session Manager to authorize yourself to display applications from other nodes.

Each node, both source and destination, must be defined in each other's network node database. For example, to display applications on node HUBBUB from ZEPHYR, HUBBUB must be entered in ZEPHYR's network node database, and ZEPHYR must be defined in HUBBUB's network node database. In addition, users on ZEPHYR must be authorized in the DECwindows Session Manager to display applications on HUBBUB. See the *VMS Networking Manual* and the

VMS Network Control Program Manual for information about entering nodes in a network node database.

/TRANSPORT=transport-name

Defines the mechanism, for example, DECNET or LOCAL, that passes information between the application and the workstation. The transport mechanism is used to send input from the user to the application and output from the application to the display. If you specify the /CREATE qualifier, the default transport is DECNET.

Use the /TRANSPORT=LOCAL qualifier to optimize the performance of applications running and displaying on the same system.

Description

VMS DECwindows gives you the ability to run applications across a network. The SET DISPLAY command lets you

- Direct the output from applications running on your workstation to another workstation.
- Direct the output from applications running on *remote* processors to your workstation. Although the application runs on another processor, it looks the same as any other application running *locally* on your workstation.

By running applications on a remote processor for local display on your workstation monitor, you can take advantage of larger computers that may be better suited to a specific computing task. By default, applications running on your workstation are displayed on your workstation. While DECwindows must be available on both nodes, only the display node must be a workstation.

The SET DISPLAY command affects only those applications run from the process from which you enter the command. This means that although you may be running one application on your workstation and displaying it on another workstation, you can continue to run applications on your workstation for display in other windows on your workstation as illustrated in Figure 1-1.

To initially direct application output to another DECwindows workstation, enter the following command:

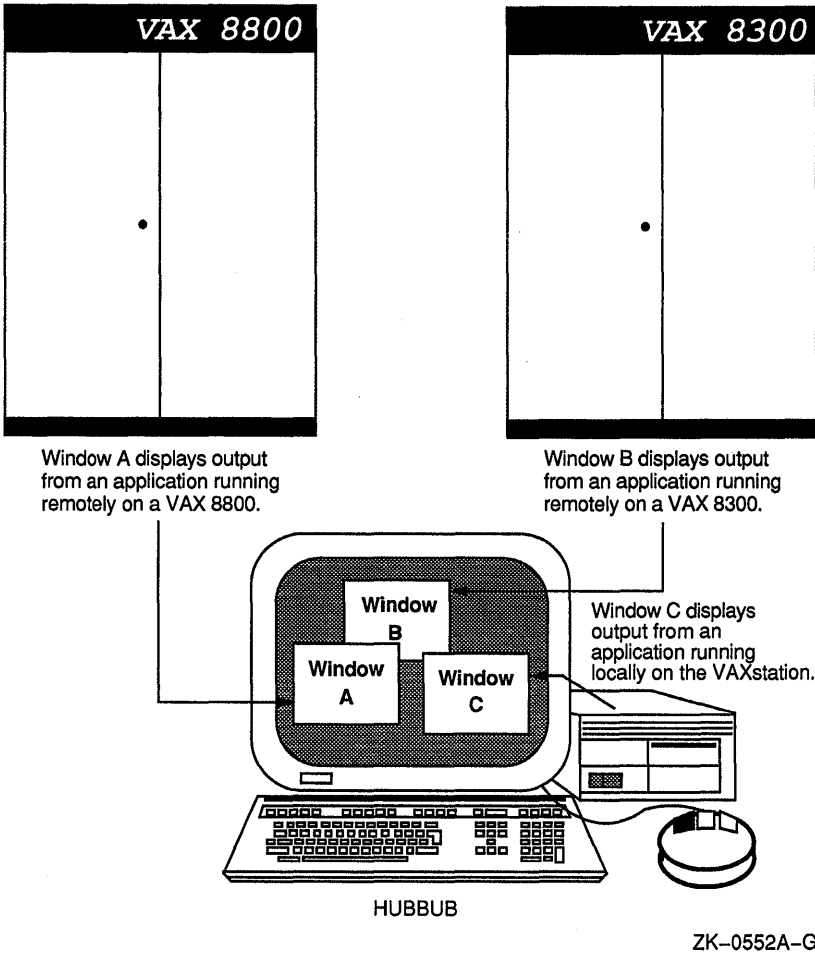
```
$ SET DISPLAY/CREATE/NODE=workstation_display/TRANSPORT=DECNET
```

Subsequently, you can direct the display to other workstation devices by entering the following command:

```
$ SET DISPLAY/NODE=workstation_display
```

1-6 General User Features: DCL Commands
SET DISPLAY Command

Figure 1-1: Running Remote and Local Applications



By default, you are authorized to run applications only from your workstation. You cannot log in to another node and direct applications to display on your workstation monitor unless you explicitly authorize yourself to do so. This prevents unauthorized users on other nodes in the network from directing output to or receiving input from your workstation without your specific permission.

Make sure the node name you use in the `SET DISPLAY` command matches the node name from which you are authorized to display applications. For example, if you specify `SET DISPLAY/CREATE/NODE=HUBBUB` from your node `ZEPHYR` but are not authorized to display applications on the DECwindows workstation `HUBBUB`, DECwindows reports that you are not authorized to use

that display. See the *VMS DECwindows User's Guide* for information about displaying applications on other workstation nodes and using the DECwindows Session Manager to authorize remote connections.

See also the description of the SHOW DISPLAY command for more information.

Example

In the following example, you are logged in to your workstation, here referred to as node 0. (0 is the standard shorthand notation for representing your node.) You want to run the DECwindows Clock application on your workstation and display it on another workstation, ZEPHYR.

Assuming you are authorized to display applications on ZEPHYR, you redirect the application's output to ZEPHYR with the SET DISPLAY command and enter the SHOW DISPLAY command to verify the location of the redirected display. You then run Clock. When you finish running Clock, you disable the redirected display by entering the SET DISPLAY/NOPERMANENT command. Finally, you enter the SHOW DISPLAY command to verify that any applications subsequently run on your node will be displayed there.

Note that a new workstation display device, WAS2, is created when you enter the SET DISPLAY/CREATE command. When you cancel the redirected display with the SET DISPLAY/NOPERMANENT command, application output is once again displayed on the workstation display device referred to by WSA1.

```
$ SHOW DISPLAY
Device:      WSA1:
Node:        0
Transport:   LOCAL
Server:      0
Screen:      0

$ SET DISPLAY/CREATE/NODE=ZEPHYR
$ SHOW DISPLAY
Device:      WSA2:
Node:        ZEPHYR
Transport:   DECNET
Server:      0
Screen:      0
$ SPAWN/NOWAIT/INPUT=NL: RUN SYS$SYSTEM:DECW$CLOCK

$ SET DISPLAY/NOPERMANENT

$ SHOW DISPLAY
Device:      WSA1:
Node:        0
Transport:   LOCAL
Server:      0
Screen:      0
```

In the following example, you are logged in to your node and want to direct the output from applications to several workstation displays in the same session. By specifying different logical names in the SET DISPLAY command, you can direct the output without changing the logical name definition for DECW\$DISPLAY.

1-8 General User Features: DCL Commands

SET DISPLAY Command

This allows you to display the output from most applications on your default display but occasionally display output on another workstation. You can also continue to run and display applications on your node. In this example, Clock is displayed on node FLOPSY, Calendar is displayed on node ZEPHYR, and Bookreader is displayed on your workstation.

Note that to run your applications with the DCL command RUN/DETACHED, you must use the device name that equates to the logical display device name you specified in the SET DISPLAY command. Use the SHOW DISPLAY command to obtain this device name.

```
$ SET DISPLAY/CREATE/NODE=FLOPSY RABBIT
$ SHOW DISPLAY RABBIT

Device:    WSA2:
Node:      FLOPSY
Transport: DECNET
Server:    0
Screen:    0

$ RUN/DETACHED/OUTPUT=WSA2: SYSS$SYSTEM:DECW$CLOCK

$ SET DISPLAY/CREATE/NODE=ZEPHYR ZNODE
$ SHOW DISPLAY ZNODE

Device:    WSA3:
Node:      ZEPHYR
Transport: DECNET
Server:    0
Screen:    0

$ RUN/DETACHED/OUTPUT=WSA3: SYSS$SYSTEM:DECW$CALENDAR

$ RUN SYSS$SYSTEM:DECW$BOOKREADER
$ SHOW DISPLAY

Device:    WSA1:
Node:      0
Transport: LOCAL
Server:    0
Screen:    0
```

1.3 SHOW DISPLAY Command

Use the SHOW DISPLAY command to see where the output from a VMS DECwindows application will be displayed.

Specify the SHOW DISPLAY command in the following format:

SHOW DISPLAY [*display-device*]

Parameter

display-device

Refers to the display-device parameter specified with the SET DISPLAY command. If you are directing application output to multiple workstations in the same session, you can use different logical names to point to each device. Using the SHOW DISPLAY command, you can specify this logical name as the display-device parameter to see where application output will be displayed.

If you do not specify a display-device string, the logical name DECW\$DISPLAY is used.

Description

DECwindows gives you the ability to run applications across a network by allowing you to

- Direct the output from applications running on your workstation to another workstation.
- Direct the output from applications running on *remote* processors to your workstation. Although the application runs on another processor, it looks the same as any other application running *locally* on your workstation.

By running applications on a remote processor for local display on your workstation, you can take advantage of larger computers that might be better suited to a specific computing task. By default, applications running on your workstation are displayed on your workstation.

You use the SET DISPLAY command to direct the output from applications to other workstations. The SHOW DISPLAY command lets you see where the output from these applications will be displayed.

Sample output from the SHOW DISPLAY command looks like this:

```
Device:   WSA2:  
Node:    0  
Transport: LOCAL  
Server:  0  
Screen:  0
```

Device is your workstation device. A new WSA*n* device is created each time you use the SET DISPLAY/CREATE command. *Node* is the network system on which the output from applications is displayed. When you are running and displaying applications on your node, *Node* is 0, which is the standard shorthand notation for representing your node. *Transport* refers to the mechanism, for example, DECNET or LOCAL, that passes information between the application and the workstation. The transport mechanism is used to send input from the user to the application and output from the application to the display. *Server* and *Screen* are 0.

1-10 General User Features: DCL Commands

SHOW DISPLAY Command

On DECwindows workstations, the Session Manager creates a default workstation device for use by DECwindows processes (such as DECterm). When you use the SET HOST command to connect to a remote node, no workstation device is created for that process and DECW\$DISPLAY is not defined. You must specifically create new display devices with the SET DISPLAY/CREATE command.

If no definition for DECW\$DISPLAY exists, entering the SHOW DISPLAY command returns an error.

See the description of the SET DISPLAY command for more information.

Example

In the following example, you are logged in to your workstation, here referred to as node 0. You want to run the DECwindows Clock application on your workstation for display on another node, ZEPHYR.

Assuming you are authorized to display applications on ZEPHYR, you redirect the application's output to ZEPHYR with the SET DISPLAY command and enter the SHOW DISPLAY command to verify the location of the redirected display. You then run Clock. When you finish running Clock, you disable the redirected display by entering the SET DISPLAY/NOPERMANENT command. Finally, you enter the SHOW DISPLAY command to verify that any applications subsequently run on your node will be displayed there.

Note that a new workstation display device, WAS2, is created when you enter the SET DISPLAY/CREATE command. When you cancel the redirected display with the SET DISPLAY/NOPERMANENT command, application output is once again displayed on the workstation display device referred to by WSA1.

```
$ SHOW DISPLAY
Device:      WSA1:
Node:        0
Transport:   LOCAL
Server:      0
Screen:      0

$ SET DISPLAY/CREATE/NODE=ZEPHYR
$ SHOW DISPLAY
Device:      WSA2:
Node:        ZEPHYR
Transport:   DECNET
Server:      0
Screen:      0

$ SPAWN/NOWAIT/INPUT=NL: RUN SYSS$SYSTEM:DECW$CLOCK

$ SET DISPLAY/NOPERMANENT

$ SHOW DISPLAY
Device:      WSA1:
Node:        0
Transport:   LOCAL
Server:      0
Screen:      0
```


1.4 DISMOUNT Qualifier

VMS Version 5.2 includes a new qualifier to the DISMOUNT command: /OVERRIDE=CHECKS. The OVERRIDE=CHECKS qualifier marks a Files-11 volume for dismounting even if files are open on the volume. If you specify /OVERRRIDE=CHECKS, DISMOUNT displays messages indicating any open files or other conditions that might have prevented dismounting, immediately followed by a message that indicates the volume has been marked for dismounting.

A substantial amount of time can pass between the time you enter DISMOUNT/OVERRIDE=CHECKS and the completion of the dismount. Always wait for the dismount to complete before you remove the volume. To verify that the dismount has completed, enter the SHOW DEVICES command.

Note that the final phase of volume dismounting occurs in the file system, and all open files on the volume must be closed before the actual dismount can be done. Also note that the file system cannot dismount a volume while any known file lists associated with the volume contain entries.

1.5 LOGIN Procedure Qualifier

VMS Version 5.2 includes a new qualifier for the LOGIN procedure: /NEW_PASSWORD. When you specify the /NEW_PASSWORD qualifier, you must change the account's password as you log in (as if the password had expired).

You can use this qualifier as a shortcut for changing your password. You might also want to use this qualifier if you suspect that your password has been discovered by an unauthorized person.

1.6 SET AUDIT Command

The SET AUDIT command includes many new qualifiers that relate to new security features. Following is complete documentation of the SET AUDIT command.

Use the SET AUDIT comand to enable or disable security auditing, modify the characteristics of the audit server process, set up long-term journaling (**archiving**) of audit events, and monitor resource consumption on the system. (Note that you must specify the /ALARM qualifier when enabling or disabling security auditing and when using the /FAILURE_MODE qualifier.)

You must hold the SECURITY privilege to use SET AUDIT.

Specify SET AUDIT in the following format:

SET AUDIT

1-12 General User Features: DCL Commands

SET AUDIT Command

Event Definition Qualifiers

/ALARM

Causes alarm messages to be sent to all terminals enabled as security operators. See the description of the DCL command **REPLY/ENABLE** for details on how to enable terminals as security operators. The **/ALARM** qualifier is required when enabling or disabling security auditing with the **/ENABLE** or **/DISABLE** qualifiers, or when specifying a failure mode with the **/FAILURE_MODE** qualifier.

/DISABLE=(keyword[,...])

Disables security auditing for the specified events. To disable alarms for all events, specify the keyword **ALL**. You can also specify the appropriate keywords to selectively disable alarms for from one to all events that are currently enabled. You must specify at least one keyword. See the **/ENABLE** qualifier description for a list of the keywords to use with the **/DISABLE** qualifier.

/ENABLE=(keyword[,...])

Enables security auditing for the specified events. To enable alarms for all events, specify the keyword **ALL**. You can also specify the appropriate keywords to selectively enable alarms for from one to all events that are currently enabled. You must specify at least one keyword.

The possible events that may be specified in the keyword list of either the **/ENABLE** or **/DISABLE** qualifier are as follows:

ACL	An event requested by an access control list (ACL) item, including ACLs on files and global sections.
ALL	All possible events.
AUTHORIZATION	The modification of any portion of the system user authorization file (SYSUAF) or network proxy authorization file (NETPROXY), including any password changes; the modification of any portion of the rights database (RIGHTSLIST).
BREAKIN=(keyword[,...])	The occurrence of one or more of the following classes of break-in attempts, as specified by one or more of the keywords: ALL , DETACHED , DIALUP , LOCAL , NETWORK , REMOTE .

SET AUDIT Command

FILE_ACCESS=(keyword[,...])	The occurrence of file and global section access events (regardless of the value specified in the object's access control list, if any). You can specify one or more of the following keywords to describe the object access event to be noted. ALL BYPASS [:access [,access...]] FAILURE [:access [,access...]] GRPPRV [:access [,access...]] READALL [:access [,access...]] SUCCESS [:access [,access...]] SYSPRV [:access [,access...]] Most of the keywords permit you to define the type of object access that was obtained with the following keywords: ALL, READ, WRITE, EXECUTE, DELETE, CONTROL.
INSTALL	The occurrence of any INSTALL operations.
LOGFAILURE=(keyword[,...])	The occurrence of one or more of the following classes of login failure, as specified by one or more of the keywords: ALL, BATCH, DETACHED, DIALUP, LOCAL, NETWORK, REMOTE, SUBPROCESS.
LOGIN=(keyword[,...])	The occurrence of one or more of the following classes of login attempts, as specified by one or more of the keywords: ALL, BATCH, DETACHED, DIALUP, LOCAL, NETWORK, REMOTE, SUBPROCESS.
LOGOUT=(keyword[,...])	The occurrence of one or more of the following classes of logouts, as specified by one or more of the keywords: ALL, BATCH, DETACHED, DIALUP, LOCAL, NETWORK, REMOTE, SUBPROCESS.
MOUNT	The issuance of a MOUNT or DISMOUNT request.

/FAILURE_MODE[=keyword]

Specifies how the VMS operating system proceeds following a failed attempt to write a security alarm to the Operator Communication Facility's (OPCOM) mailbox. Specify one of the following keywords with the **/FAILURE_MODE** qualifier:

Option	Description
WAIT	Indicates that processes are placed in the MWAIT state to wait until the resource is available. This is the default.
IGNORE	Indicates that failing security alarms are to be ignored. The first failed alarm causes an error message to be written to the operator console and log file. The system maintains a count of the lost alarms, which can be displayed with SHOW AUDIT.
CRASH	Forces a system failure if security alarms cannot be written.

The **/ALARM** qualifier is required when specifying an audit failure mode.

1-14 General User Features: DCL Commands

SET AUDIT Command

/VERIFY (default)

/NOVERIFY

Specifies that control is not returned to the user (at the DCL command level) until the audit server has completed the request.

Audit Journal Qualifiers

/DESTINATION=file-spec

Specifies the name and location of the security audit log file in the audit server database. The device, if part of the file specification, must be a disk volume. Because the system security log file is automatically created when the system is first installed and restored each time the system boots, this qualifier is required only when you want to move the log file.

Once you have updated the audit server database, execute the command SET AUDIT/SERVER=NEW_LOG to make the new location of the log file known to all audit server processes in the cluster. The previous audit log file is closed, and all subsequent audit event messages generated on the cluster are redirected to the new audit log file.

The */JOURNAL=SECURITY* qualifier is required when redirecting the system security audit log file with the */DESTINATION* qualifier.

/JOURNAL[=journal-name]

Specifies the name of the audit journal. The default, */JOURNAL=SECURITY*, represents the system security audit log file and is currently the only supported audit journal type. The */JOURNAL* qualifier is required when changing the location of the audit log file with the */DESTINATION* qualifier.

Audit Server Qualifiers

/INTERVAL=(option-keyword[,...])

Specifies the delta times to be used for regular audit server operations. See the *VMS DCL Concepts Manual* for information about specifying delta times.

In most cases, the defaults noted should be sufficient.

Option Keyword	Description
ARCHIVE_FLUSH=time	Specifies the period of time the audit server waits before flushing information to be archived. The default is 1 minute.
JOURNAL_FLUSH=time	Specifies the period of time the audit server waits before flushing information in the various audit journal buffers. The default is 5 minutes.

Option Keyword	Description
RESOURCE_SCAN=time	Specifies the period of time the audit server waits before monitoring the volume containing the audit journal for resource exhaustion. Resource exhaustion occurs when the volume has no free disk space. The default is 5 minutes.
RESUME_SCAN=time	Specifies the period of time the audit server waits before reviewing an existing resource exhaustion condition. The default is 15 minutes.

/LISTENER=device
/NOLISTENER

Specifies the name of a mailbox device that will receive a copy of all security audit events. The user-defined mailbox can be used for processing of system security events as they occur, rather than logging events to the system audit journal for inspection at a future time. See the *VMS Audit Analysis Utility Manual* for a description of the message formats written to the listener mailbox.

Specify the SET AUDIT/NOLISTENER command to remove a listener device from the system.

/SERVER=option-keyword[,...]

Specifies the audit server characteristics to be modified.

In most cases, the defaults noted should be sufficient.

Option Keyword	Description
CREATE_SYSTEM_LOG	Causes the audit server to create a new local system security audit log file. Other audit servers in the cluster are not affected. This keyword may be used by sites operating a multiple-environment cluster where it may be necessary to create a new log file on a specific node in the cluster. CREATE_SYSTEM_LOG is synonymous with NEW_LOG for nonclustered systems.
EXIT	Initiates an audit server shutdown. This is the only method for removing the audit server process from the system; the audit server cannot be deleted or suspended.

1-16 General User Features: DCL Commands

SET AUDIT Command

Option Keyword	Description
FINAL_ACTION=action	<p>Specifies the action taken by the audit server when resource exhaustion conditions have been met. Resource exhaustion occurs when the audit server attempts to buffer audit messages and runs out of virtual memory. (See the <i>Guide to VMS System Security</i> for more information about resource monitoring.) Specify one of the following values:</p> <p>CRASH Crash the system if the system runs out of virtual memory. This is the default.</p> <p>IGNORE_NEW Ignore new event messages until resources are available. Event messages leading up to the resource condition are saved; new messages are lost.</p> <p>PURGE_OLD Removes old event messages until resources are available in order to save the most current messages.</p>
FLUSH	Copies all buffered audit and archive records to the audit log file and security archive file, respectively.
NEW_LOG	Creates a new clusterwide audit log file. The audit log file is created by the audit server process running on the local system and is opened by all audit servers in the cluster. (Typically, this is used daily to generate a new version of the audit log file.)
REDIRECT_SYSTEM_LOG	Causes the audit server on the local node to redirect security event messages to a new audit log file, whose location was previously defined by the /DESTINATION qualifier. Audit server processes (and log files) on other nodes in the cluster are unaffected.
RESUME	Requests the audit server process to resume normal activity on the system, if adequate disk space is available. Normally, once a resource monitoring action threshold has been reached, the audit server process suspends most system activity and waits 15 minutes before attempting to resume normal system activity.
START	Starts the audit server process on the system.

/VERIFY (default)

/NOVERIFY

Specifies that control is not returned to the user (at the DCL command level) until the audit server has completed the request.

Archiving Qualifiers

/ARCHIVE=option-keyword[,...]

Specifies the classes of audit events to be written to the security archive file. Specify one or more of the following keywords:

Option Keyword	Description
NONE	Disables archiving on the system. By default, archiving is disabled on the system.
[NO]ALL	Enables or disables archiving of all system security events.
SYSTEM_ALARM	Enables archiving of all system-generated alarm events.
SYSTEM_AUDIT	Enables archiving of all system-generated audit events. Reserved for future use.
USER_ALARM	Enables archiving of all user-generated alarm events. Reserved for future use.
USER_AUDIT	Enables archiving of all user-generated audit events. Reserved for future use.

/DESTINATION=file-spec

Specifies the name of the archive log file. Events may be archived to a local or remote file on any file-structured disk device. See the *Guide to VMS System Security* for information about creating a security archive file.

/VERIFY (default)

/NOVERIFY

Specifies that control is not returned to the user (at the DCL command level) until the audit server has completed the request.

Resource Monitoring Qualifiers

/[NO]EXCLUDE=process-id

Adds a process ID (PID) to the audit server process exclusion list. The process exclusion list contains those processes that will not be suspended by the audit server process if a resource exhaustion reaches the **action** threshold. By default, the following processes are always contained in the process exclusion list and are never candidates for process suspension:

CACHE_SERVER
 CLUSTER_SERVER
 CONFIGURE
 JOB_CONTROL
 OPCOM
 SWAPPER
 VWS\$DISPLAYMGR
 VWS\$EMULATORS

1-18 General User Features: DCL Commands

SET AUDIT Command

Use the SET AUDIT/NOEXCLUDE=*process-id* command to remove a process from the process exclusion list. (PIDs are not automatically removed from the process exclusion list when processes log out from the system.)

/JOURNAL[=*journal-name*]

Specifies the name of the audit journal. The default, /JOURNAL=SECURITY, represents the system security audit log file and is currently the only supported audit journal type. The /JOURNAL qualifier is required when specifying resource monitoring characteristics with the /RESOURCE or /THRESHOLD qualifiers.

/RESOURCE=*option-keyword*[,...]

Controls whether resource monitoring is in effect on the system, specifies the method used to monitor available resources, and determines the action the audit server will take if the available resources are depleted. The /JOURNAL qualifier is required when specifying resource monitoring with the /RESOURCE qualifier. See the *Guide to VMS System Security* for more information about resource monitoring.

Option Keyword	Description
DISABLE	Disables or enables resource monitoring on the security audit journal file. By default, resource monitoring is enabled on the system.
ENABLE	
MONITOR_MODE= <i>mode</i>	Specifies the method the audit server uses to monitor available resources. Specify one of the following keywords:
COUNT	Controls whether resource monitoring is based on the amount of free disk space required to store a fixed number of event messages.
PERCENTAGE	Controls whether resource monitoring is based on the percentage of the disk volume or volume set available.
SPACE	Controls whether resource monitoring is based on the number of free blocks on the disk. This is the default method used for resource monitoring.
TIME	Controls whether resource monitoring is based on the amount of free disk space needed to store events that occur over a fixed period of time (in seconds).

/THRESHOLD=*type*

Specifies the thresholds the audit server uses for resource monitoring. The values that may be specified for each of the thresholds described depends on the mode of resource monitoring enabled on the system (see /RESOURCE=MONITOR_MODE). The /JOURNAL qualifier is required when modifying audit server thresholds with the /THRESHOLD qualifier.

Threshold Type	Meaning
WARNING=value	Specifies the threshold at which the audit server notifies all security operator terminals that resource exhaustion has occurred.
ACTION=value	Specifies the threshold at which the audit server suspends normal system activity.
RESUME=value	Specifies the threshold at which the audit server resumes normal system activity.

The following table lists the default warning, action, and resume thresholds for each resource monitor mode. Normally, the defaults listed should be sufficient.

Monitor Mode	Resource Monitoring Thresholds		
	WARNING	ACTION	RESUME
SPACE (blocks)	1000	250	750
PERCENTAGE (of volume)	1	0	1
COUNT (number of messages)	5000	1250	3750
TIME (seconds)	1000	250	750

/VERIFY (default)

/NOVERIFY

Specifies that control is not returned to the user (at the DCL command level) until the audit server has completed the request.

Description

The SET AUDIT command enables security auditing to send alarms to terminals that have been enabled as security operators and to make an entry in the system security audit log file whenever specified events are detected by the system. For example, you can use a SET AUDIT command to cause an alarm message to occur if break-in attempts are observed or if the system user authorization file (SYSUAF) or network proxy file is modified.

The primary purpose of the system security audit log file is to capture details about who or what performed an action that might have significance for system security and at what time. From this historical record, it may be possible to draw conclusions about attacks on the system and how to counteract them. The procedures for properly handling and maintaining the audit log files are described in the *VMS Audit Analysis Utility Manual*. The log files can be inspected and their data analyzed with the DCL command ANALYZE/AUDIT.

1-20 General User Features: DCL Commands

SET AUDIT Command

In addition to enabling or disabling security auditing on the system, the SET AUDIT command can be used to change the default characteristics of the audit server, set up long-term journaling (archiving) of audit events, and monitor resource consumption on the system. See the *Guide to VMS System Security* for information about the audit server, the security archive file, and resource monitoring.

To display the results of a SET AUDIT command, enter the DCL command SHOW AUDIT/ALL.

Examples

```
$ SET AUDIT/ARCHIVE=ALL -  
_ $ /DESTINATION=AUDIT$: [SECURITY_HISTORY]ARCHIV_001.DAT
```

The SET AUDIT command in this example enables archiving of system security event messages to a file named ARCHIV_001.DAT in the directory AUDIT\$: [SECURITY_HISTORY].

Note that /ARCHIVE=ALL is synonymous with /ARCHIVE=SYSTEM_ALARM and is the only type of archiving currently supported.

```
$ SET AUDIT/JOURNAL=SECURITY -  
_ $ /DESTINATION=AUDIT$: [AUDIT]SECURITY_AUDIT.LOG  
$ SET AUDIT/SERVER=NEW_LOG
```

The first SET AUDIT command in this example updates the audit server database with the new name and location of the system security audit log file. The second command in the example causes all audit server processes in the cluster to open the new log file.

```
$ SET AUDIT/ALARM/ENABLE=ALL/DISABLE=FILE:ALL
```

The SET AUDIT command in this example enables all classes of security events except file access alarms.

1.7 SET FILE Qualifier: /[NO]SEMANTICS

For VMS Version 5.2, the SET FILE qualifier /[NO]SEMANTICS enables you to tag a Digital Document Interchange Format (DDIF) file through the DCL interface for VMS systems. You use /SEMANTICS=semantics-tag to create or change a semantics tag. You use /NOSEMANTICS to remove a semantics tag from a file.

The following command line tags the file X.DDIF as a DDIF file by assigning the appropriate value to the /SEMANTICS qualifier:

```
$ SET FILE X.DDIF/SEMANTICS=DDIF
```

For more information about file tags, see Section A.1 in Appendix A. For more information about the SET FILE command, see the *VMS DCL Dictionary*.

1.8 SHOW AUDIT Command

The SHOW AUDIT command includes new qualifiers that relate to new security features. Following is complete documentation of the SHOW AUDIT command.

Use the SHOW AUDIT command to display the security auditing characteristics in effect on the system. You must hold the SECURITY privilege to use SHOW AUDIT.

Specify SHOW AUDIT in the following format:

SHOW AUDIT

Qualifiers

/ALL

Displays all available auditing information including the following: name and location of the system security audit log file; type of security events enabled on the system; action the system will take if an attempt to write an audit event message fails (failure mode); name and location of the security archive file; information about the audit server process, such as the action taken if the audit server process runs out of virtual memory.

/ALARM

Displays the security events currently enabled on the system.

/ARCHIVE

Displays the name and location of the security archive file (if enabled).

/FAILURE_MODE

Displays the failure mode currently in effect on the system.

/JOURNAL

Displays the name and location of the system security audit log file.

/OUTPUT[=file-spec]

/NOOUTPUT

Controls where the output of the command is sent. If you do not enter the qualifier, or if you enter /OUTPUT without a file specification, the output is sent to the current process default output stream or device, identified by the logical name SYS\$OUTPUT.

/SERVER

Displays information about the audit server process. Currently, the only information displayed is the action the audit server takes if all attempts to allocate process virtual memory are unsuccessful.

1-22 General User Features: DCL Commands

SHOW AUDIT Command

Description

The SHOW AUDIT command displays the set of features that have been enabled for auditing with the DCL command SET AUDIT. If no auditing has been enabled, the display reports briefly that security alarms are currently disabled. The display is directed to the current SYS\$OUTPUT device.

If you do not specify any qualifiers with the SHOW AUDIT command, only the auditing failure mode and currently enabled security events are displayed.

It is useful to check which auditing features are enabled whenever you plan to add or delete features with a subsequent SET AUDIT command.

Example

```
$ SHOW AUDIT/ALL
List of audit journals:
  Journal name:          SECURITY
  Journal owner:        (system audit journal)
  Destination:          SYS$COMMON:[SYSMGR]AUDIT.AUDIT$JOURNAL
  Monitoring:           free disk space
  Warning threshold:    1000 blocks
  Action threshold:     250 blocks
  Resume threshold:     750 blocks

Security auditing server characteristics:
  Final resource action: crash system

Security archiving information:
  Archiving events:     system audits, system alarms
  Archive destination:  SPAR::SYS$MANAGER:K9.AUDIT$JOURNAL

Security alarm failure mode is set to:
  WAIT                 Processes will wait for resource

Security alarms currently enabled for:
  ACL
  AUTHORIZATION
  BREAKIN:             (DIALUP, LOCAL, REMOTE, NETWORK, DETACHED)
  LOGIN:               (BATCH, DIALUP, LOCAL, REMOTE, NETWORK, SUBPROCESS, DETACHED)
  LOGFAILURE:          (BATCH, DIALUP, LOCAL, REMOTE, NETWORK, SUBPROCESS, DETACHED)
```

The SHOW AUDIT command in this example displays all the available security auditing information, including the following:

- The name and location of the system security audit log file.
- The method used by the audit server to monitor available resources on the system and the threshold values at which the audit server sends out notification or suspends or resumes activity on the system.
- Information about the audit server process.
- The name and location of the security archive file. By default, archiving is not enabled on the system.

- The security alarm failure mode in effect on the system.
- The classes of security events enabled on the system. All security events generated are written to the system security audit log file and broadcast as alarm messages to all security operator terminals.

1.9 SHOW SYSTEM Qualifiers

VMS Version 5.2 includes two new qualifiers to the SHOW SYSTEM command: /CLUSTER and /NODE[=(name,...)].

The /CLUSTER qualifier displays all processes on all nodes in a VAXcluster.

The /NODE[=(name,...)] qualifier displays all processes on the specified node or nodes in a VAXcluster. The qualifier /NODE without a value displays all processes on the local node.

For more information about using SHOW SYSTEM, see the *VMS DCL Dictionary*.

1.10 SHOW USERS Command

VMS Version 5.2 includes several new qualifiers to the SHOW USERS command. New SHOW USERS qualifiers fall into one of the following types:

- Qualifiers that display information about specific categories of users (for example, interactive users)
- Qualifiers that display information about users in a cluster

The following table lists the qualifiers and their functions:

Feature	Function
/BATCH	Displays all batch users in the system.
/CLUSTER	Displays all users on all nodes in a VAXcluster.
/FULL	Displays complete information, rather than the default summary, about each process.
/INTERACTIVE	Displays all interactive users in the system.
/NETWORK	Displays all network users in the system.
/NODE[=name(...)]	Displays all users on the specified node or nodes in a VAXcluster. The qualifier /NODE without a value displays all users on the local node.
/SUBPROCESS	Displays all subprocess users in the system.

For more information about using SHOW USERS, see the *VMS DCL Dictionary*.

1.11 Lexical Function: F\$CONTEXT

F\$CONTEXT specifies selection criteria for use with the F\$PID function. The F\$CONTEXT function enables the F\$PID function to obtain information about processes from any node in a VAXcluster.

The F\$CONTEXT function is called as many times as desired to produce the criteria needed; however, each call can specify only one selection item. Lists of item values are allowed, where appropriate, and more than one context can be operated upon at a time.

After establishing the selection criteria with appropriate calls to F\$CONTEXT, F\$PID is called repeatedly to return all the process identification (PID) numbers that meet the criteria specified in the F\$CONTEXT function. When there are no more such processes, the F\$PID function returns a null string.

After the F\$PID function is called, the context symbol is considered “frozen”; F\$CONTEXT cannot be called again with the same context symbol until the associated context selection criteria have been deleted. If you attempt to set up additional selection criteria with the same context-symbol, an error message is displayed. However, the context and selection criteria are not affected and calls to F\$PID can continue.

The F\$CONTEXT function uses process memory to store the selection criteria. This memory is deleted under two circumstances. Memory is deleted when F\$PID is called and a null string is returned, that is, when all processes that meet the selection criteria have been returned. Memory also is deleted if the CANCEL selection-item keyword is used in a call to F\$CONTEXT with an established context. This type of call is appropriate for a Ctrl/Y or another condition handling routine.

The following sections provide reference information about the F\$CONTEXT lexical function.

Format

F\$CONTEXT (*context-type, context-symbol, selection-item, selection-value, value-qualifier*)

Return Value

A null string.

ARGUMENTS

context-type

The type of context to be built. The following table shows the valid context type and the lexical function for which a context is built:

Lexical Function: F\$CONTEXT

Context Type	Description
PROCESS	For use in constructing selection criteria for F\$PID

context-symbol

A symbol that DCL uses to refer to the context memory being constructed by F\$CONTEXT. F\$PID uses this context symbol to process the appropriate list of PIDs.

The first time you use the F\$CONTEXT function in a command procedure, use a symbol that is either undefined or equated to the null string (""). The symbol created will be a local symbol of type "PROCESS_CONTEXT". When the context is no longer valid—that is, when all PIDs have been retrieved by F\$PID calls or an error occurs during one of these calls—the symbol no longer has a type of "PROCESS_CONTEXT". Then you can use the F\$TYPE function in the command procedure to find out if it is necessary to cancel the context.

After setting up the selection criteria, use this context symbol when calling F\$PID.

selection-item

A keyword that tells F\$CONTEXT which selection criteria to use. Use only one selection-item keyword per call to F\$CONTEXT.

The following is a table of valid selection-item keywords for the PROCESS context-type:

Selection Item	Selection Value	Value Qualifiers	Comments
ACCOUNT	string	EQL, NEQ	Valid account name or list of names. Wildcard characters are acceptable.
AUTHPRI	integer	GEQ, GTR, LEQ, LSS, EQL, NEQ	Valid authorized base priority (0-31).
CANCEL			This keyword cancels the selection criteria for this context.
CURPRIV	keyword	ALL, ANY, EQL, NEQ	Valid privilege name keyword or list of keywords. For more information, see the <i>VMS DCL Concepts Manual</i> .
GRP	string	GEQ, GTR, LEQ, LSS, EQL, NEQ	Group number or name.
HW_MODEL	integer	EQL, NEQ	Valid hardware model number.

1-26 General User Features: DCL Commands
 Lexical Function: F\$CONTEXT

Selection Item	Selection Value	Value Qualifiers	Comments
HW_NAME	string	EQL, NEQ	Valid hardware name or a list of keywords. Wildcard characters are acceptable.
JOBPRCNT	integer	GEQ, GTR, LEQ, LSS, EQL, NEQ	Number of subprocesses for the entire job.
JOBTYP	keyword	EQL, NEQ	Valid jobtype keyword. Valid keywords are DETACHED, NETWORK, BATCH, LOCAL, DIALUP, and REMOTE. For more information, see the <i>VMS DCL Concepts Manual</i> .
MASTER_PID	string	EQL, NEQ	PID of the master process.
MEM	string or integer	GEQ, GTR, LEQ, LSS, EQL, NEQ	UIC member number or name.
MODE	keyword	EQL, NEQ	Valid process mode keyword. Valid keywords are OTHER, NETWORK, BATCH, and INTERACTIVE. For more information, see the <i>VMS DCL Concepts Manual</i> .
NODE_CSID	integer	EQL, NEQ	Node's cluster ID number.
NODENAME	string	EQL, NEQ	Node name or list of node names. Wildcard characters are acceptable. The default value is your local node. To request all nodes, use an asterisk.
OWNER	string	EQL, NEQ	PID of immediate parent process.
PRCNT	integer	GEQ, GTR, LEQ, LSS, EQL, NEQ	Subprocess count of process.
PRCNAM	string	EQL, NEQ	Process name or list of process names. Wildcard characters are acceptable.
PRI	integer	GEQ, GTR, LEQ, LSS, EQL, NEQ	Process priority level number (0-31).
PRIB	integer	GEQ, GTR, LEQ, LSS, EQL, NEQ	Base process priority level number (0-31).
STATE	keyword	EQL, NEQ	Valid process state keyword. For more information, see the description of the \$GETJPI service in the <i>VMS System Services Reference Manual</i> .
STS	keyword	EQL, NEQ	Valid process status keyword. For more information, see the description of the \$GETJPI service in the <i>VMS System Services Reference Manual</i> .

Selection Item	Selection Value	Value Qualifiers	Comments
TERMINAL	string	EQL, NEQ	Terminal name or list of names. Wildcard characters are acceptable.
UIC	string	EQL, NEQ	UIC identifier (that is, of the form "[group,member]").
USERNAME	string	EQL, NEQ	User name or list of user names. Wildcard characters are acceptable

selection-value

Value of the selection criteria. For example, to process all the processes running on node MYVAX, specify "MYVAX" with the "NODENAME" keyword. For example:

```
$ X = F$CONTEXT("PROCESS", ctx, "NODENAME", "MYVAX", "EQL")
```

Values that are lists are valid with some selection items. The items must be separated by a comma. For example, to specify a list of the nodes MYVAX, HERVAX, and HISVAX, specify the following:

```
$ X = F$CONTEXT("PROCESS", ctx, "NODENAME", "MYVAX,HERVAX,HISVAX", "EQL")
```

You can use wildcard characters for some values. Using wildcard characters for selection items is similar to using wildcard characters for file names. Use an asterisk (*) to match zero or more characters. Use a percent sign (%) to match exactly one character.

value-qualifier

You must qualify selection values. You can qualify a number, for example, by requesting that the selection be based on the process value being less than (LSS), less than or equal to (LEQ), greater than (GTR), greater than or equal to (GEQ), equal to (EQL), or not equal to (NEQ) the value specified in the call to F\$PID.

You can qualify some lists with the ALL, ANY, EQL, or NEQ keywords. Such lists are usually masks, such as the process privilege mask, which consists of the set of enabled privileges. ALL requires that all items in the list be true for a process. ANY requests that any item in the list be part of the attributes of a process. EQL means that the values must match exactly (that is, values specified must not be true of the process). NEQ requires that the value must not match.

The difference between ALL and EQL is that the values specified with ALL must exist, but other unspecified values may exist also. EQL requires that all values specified must exist, and others must not. For example, to request those processes whose current privileges include TMPMBX and OPER, but may include others, specify the ALL keyword. To request those processes whose current privileges are TMPMBX and OPER exclusively, specify the EQL keyword.

1-28 General User Features: DCL Commands

Lexical Function: F\$CONTEXT

description

Use the F\$CONTEXT function to set up selection criteria for the F\$PID function.

EXAMPLE

```
#!Establish an error and CTRL/Y handler
$!
$ ON ERROR THEN GOTO error
$ ON CONTROL_Y THEN GOTO error
$!
$ ctx = ""
$ temp = F$CONTEXT ("PROCESS", ctx, "NODENAME", "*", "EQL")
$ temp = F$CONTEXT ("PROCESS", ctx, "USERNAME", "M*,SYSTEM", "EQL")
$ temp = F$CONTEXT ("PROCESS", ctx, "CURPRIV", "SYSPRV,OPER", "ALL")
$!
$!Loop over all processes that meet the selection criteria.
$!Print the PID and name of the image for each such process.
$!
$ loop:
$ pid = F$PID(ctx)
$ IF pid .EQS. ""
$ THEN
$     GOTO endloop
$ ELSE
$     image = F$GETJPI(pid,"IMAGENAME")
$     WRITE SYS$OUTPUT pid," - ",image
$     GOTO loop
$ ENDIF
$!The loop over the processes has ended.
$!
$ endloop:
$!
$ EXIT
$!
$!Error handler. Clean up the context's memory
$!with the CANCEL selection item keyword.
$!
$ error:
$ IF F$TYPE(ctx) .eqs. "PROCESS_CONTEXT" THEN -
temp = F$CONTEXT ("PROCESS", ctx, "CANCEL")
$!
$ EXIT
```

In this example, F\$CONTEXT is called three times to set up selection criteria. The first call requests that the search take place on all nodes in the cluster. The second call requests that only the processes whose user name either starts with an *M* or is *SYSTEM* be processed. The third call restricts the selection to those processes whose current privileges include both SYSPRV and OPER but might have other privileges set.

Lexical Function: F\$CONTEXT

The command lines inside of loop: and endloop: continually call F\$PID to obtain the processes that meet the criteria set up in the F\$CONTEXT calls. After retrieving each process identification (PID) number, F\$GETJPI is called to return the name of the image running in the process. Finally, the procedure displays the PID of the process and the name of the image.

In case of an error or Ctrl/Y, control is passed to error: and the context is closed if necessary.

In this example, note the check for the symbol type "PROCESS_CONTEXT". If the symbol has this type, selection criteria must be canceled by a call to F\$CONTEXT. If the symbol is not of the type "PROCESS_CONTEXT", either selection criteria have not been set up yet in F\$CONTEXT, or the symbol has been used with F\$PID until an error occurred or until the end of the process list was reached.

For more information about the F\$CONTEXT function, see the *VMS DCL Dictionary*.

1.12 Lexical Function Item Codes

You use the F\$FILE_ATTRIBUTES lexical function in DCL assignment statements and expressions to return file attribute information. The following table lists the new F\$FILE_ATTRIBUTES item codes and their functions:

Lexical Function Item Codes	Function
AI	Returns TRUE or FALSE. Returns TRUE if AI journaling is enabled.
BI	Returns TRUE or FALSE. Returns TRUE if BI journaling is enabled.
ERASE	Returns TRUE or FALSE. Returns TRUE if a file's contents are erased before the file is deleted.
FFB	Returns the first free byte.
GBC	Returns the global buffer count.
JOURNAL_FILE	Returns TRUE or FALSE. Returns TRUE if the file is a journal file.
LOCKED	Returns TRUE or FALSE. Returns TRUE if a file is deaccessed-locked.
RU	Returns TRUE or FALSE. Returns TRUE if RU journaling is enabled.
STORED_SEMANTICS	Returns an ASCII string that represents stored semantics.

Chapter 2

System Management Features

This chapter describes the new system management features of VMS Version 5.2. These features enhance the following components of the operating system:

- Security
- SYSGEN Utility
- SYSMAN Utility
- License Management Facility
- NETCONFIG.COM
- Disk and Tape support
- Backup Utility
- VMS DECwindows

The following sections describe the new features in detail.

2.1 Security Auditing

VMS Version 5.2 includes new features that improve the performance and effectiveness of security auditing on a system or cluster. Changes made to the VMS operating system to support new security auditing features affect all system managers and site security administrators. Table 2-1 summarizes changes to security auditing.

2-2 System Management Features

Security Auditing

Table 2-1: New Security Auditing Features

Feature	Function
Audit server database	Maintains a list of security events and audit server characteristics enabled on the system.
Audit server process	Oversees security auditing on the system: controls logging of security events to a clusterwide security audit log file; monitors resources (virtual memory and disk space) required for security auditing; prevents the loss of security information when resources are depleted.
Audit Analysis Utility	Allows you to extract selective information from the security audit log file.

As an additional feature of the VMS Version 5.2 operating system, security auditing is enabled on all systems, and a default set of security event classes is enabled and written to the system security audit log file. (In versions of the VMS operating system prior to Version 5.2, security auditing was disabled by default.) Section 2.1.2.1 describes how to disable security auditing.

The following sections provide an overview of these new features.

2.1.1 Audit Server Database

The audit server database is the collection of information used by the audit server process to oversee security auditing on the system. The database is updated whenever security auditing characteristics are modified and restored the next time the system is rebooted.

The audit server database, `AUDIT_SERVER.DAT`, is created in the system manager directory when you first install or upgrade your system to VMS Version 5.2. `AUDIT_SERVER.DAT` initially contains the following security auditing information:

- The name and location of the system security audit log file. By default, the file is named `SECURITY_AUDIT.AUDIT$JOURNAL` and is located in the system manager directory, `SYS$COMMON:[SYSMGR]`.
- The default set of security audit event classes to be logged to the system security audit log file, including the following: `AUTHORIZATION`, `AUDIT`, and `BREAKIN`.
- Various timers associated with the operation of the audit server process.
- Information used by the audit server process to monitor the consumption of resources on the system.
- The location of the security archive file, if enabled.

- The classes of information that are logged to the security archive file.

Information contained in the audit database is updated each time you enter the SET AUDIT command. Because all changes are reflected immediately on the system and are also saved when the system is shut down, you should remove any SET AUDIT commands from your site-specific startup command procedure. For more information about the SET AUDIT and SHOW AUDIT commands, see Section 1.6 and Section 1.8.

2.1.2 Audit Server Process

The audit server process (AUDIT_SERVER) is a noninteractive, detached process created during the system bootstrap that performs the following functions:

- Creates a binary, clusterwide security audit log file
- Monitors system resources to prevent the loss of security audit messages

The addition of a single clusterwide security audit log file in VMS Version 5.2 reduces the work required of the security administrator and increases the effectiveness of the audit trail. Prior to VMS Version 5.2, all security event messages were logged to the operator log file, along with all other types of operator messages. In a VAXcluster environment, each node maintained a separate copy of the operator log file.

Additional benefits of the security audit log file include its public format, which allows third-party analysis tools to be developed, and its binary format, which reduces disk space required by the security auditing software.

To prevent the loss of information due to resource exhaustion, the audit server monitors the remaining disk space available for the new audit log file. Should disk space become critical, the audit server follows a specific list of actions that will reduce the amount of auditing information being generated. These actions can include suspending noncritical processes, ignoring security events, or shutting down the system.

The audit server process maintains, in its private database, most of the site security auditing settings and contains default settings for all new features. Nonetheless, it is necessary for the site security administrator to be able to modify certain characteristics of the audit server process. Therefore, several new qualifiers have been added to the SET AUDIT command.

A complete description of the audit server process is available in the *Guide to VMS System Security*. See Section 1.6 and Section 1.8 for descriptions of the SET AUDIT and SHOW AUDIT commands.

2-4 System Management Features

Security Auditing

2.1.2.1 Disabling the OPCOM and AUDIT_SERVER Processes

Beginning with VMS Version 5.2, the Operator Communication Facility (OPCOM) process is started by default on all VAX systems running the VMS operating system. The AUDIT_SERVER process is also started. If the physical memory or disk storage space on your system is especially limited, you might consider removing these processes from the system startup procedure with the SYSMAN Utility, as shown in the following example:

```
$ SET PROCESS/PRIVILEGES=(OPER,BYPASS)
$ RUN SYSS$SYSTEM:SYSMAN
SYSMAN> STARTUP SET DATABASE STARTUP$STARTUP_VMS
SYSMAN> STARTUP DISABLE FILE VMS$CONFIG_050_OPCOM.COM/NODE=*
SYSMAN> STARTUP DISABLE FILE VMS$CONFIG_050_AUDIT_SERVER.COM/NODE=*
SYSMAN> EXIT
```

To add the OPCOM and AUDIT_SERVER processes to the system startup procedure and restart security auditing the next time the system reboots, enter the following SYSMAN commands:

```
$ SET PROCESS/PRIVILEGES=(OPER,BYPASS)
$ RUN SYSS$SYSTEM:SYSMAN
SYSMAN> STARTUP SET DATABASE STARTUP$STARTUP_VMS
SYSMAN> STARTUP ENABLE FILE VMS$CONFIG_050_OPCOM.COM/NODE=*
SYSMAN> STARTUP ENABLE FILE VMS$CONFIG_050_AUDIT_SERVER.COM/NODE=*
SYSMAN> EXIT
```

To add OPCOM and AUDIT_SERVER to the system without rebooting, execute the following DCL commands:

```
$ @SYSS$SYSTEM:STARTUP OPCOM
$ SET AUDIT/SERVER=START
```

See the *VMS SYSMAN Utility Manual* for more information about using the SYSMAN Utility.

2.1.3 VMS Audit Analysis Utility

The new clusterwide Audit Analysis Utility, invoked with the DCL command ANALYZE/AUDIT, improves the site security administrator's ability to review the security events logged to the system. Prior to VMS Version 5.2, all security event messages enabled with the DCL command SET AUDIT were copied to the operator log file (OPERATOR.LOG in the SYS\$MANAGER directory). VMS provided the DCL command procedure SECAUDIT.COM to extract security messages from the operator log file.

Security event messages can be extracted from the system security audit log file and displayed on a terminal (or output to a file) using combinations of ANALYZE/AUDIT selection qualifiers. The Audit Analysis Utility also features an interactive, screen-oriented command mode, which can be invoked by entering Ctrl/C during the display. After you enter interactive command mode, you can specify a record number for viewing, specify different selection and exclusion criteria for viewing audit records, obtain online help, and move forward or backward, by any number of records, through the security audit log file.

For more information about the Audit Analysis Utility, see the *VMS Audit Analysis Utility Manual*.

2.2 SYSGEN Utility

The VMS Version 5.2 System Generation Utility (SYSGEN) contains the following new command and parameter:

- DEINSTALL command
- ERLBUFFERPAGES parameter

2.2.1 DEINSTALL Command Description

DEINSTALL removes or “deinstalls” system page files and system swap files. Any file that is installed with the INSTALL command can be removed with the DEINSTALL command.

Use of the DEINSTALL command requires the CMKRNL privilege.

Format

DEINSTALL *filespec*

Parameter

filespec

Specifies the name of the page or swap file. The default file type is SYS.

Qualifiers

/ALL

Deinstalls all page and swap files currently installed on the system. This command is most useful during an orderly system shutdown procedure where all disk volumes are being dismounted.

/INDEX=n

Deinstalls a page or swap file specified by the page file index. The page file index is presented in the SHOW MEMORY/FILES/FULL display as “Paging File Number.”

/PAGEFILE

Specifies that the file to be deinstalled is a page file.

/SWAPFILE

Specifies that the file to be deinstalled is a swap file.

Example

```
SYSGEN> DEINSTALL SYS$SYSTEM:PAGEFILE.SYS/PAGEFILE
```

The command in this example deinstalls the system page file.

2-6 System Management Features

SYSGEN Utility

2.2.2 ERLBUFFERPAGES Parameter

The ERLBUFFERPAGES parameter specifies the number of pages of memory to allocate for each buffer requested by the ERRORLOGBUFFERS parameter. The ERLBUFFERPAGES parameter has a default value of 2 pages and a maximum value of 32 pages. The default value of 2 pages consists of one page for each buffer greater than the previous buffer size.

2.3 SYSMAN Utility

The SYSMAN Utility has the following new features for VMS Version 5.2:

- The functions of the auto-login facility (ALF) have been added to the SYSMAN Utility.
- In clustered environments that use common SYSUAF and RIGHTSLIST databases, SYSMAN now grants privileges and rights differently.
- The CONFIGURATION SHOW CLUSTER display now includes the cluster's multicast address.
- The new commands LICENSE LOAD and LICENSE UNLOAD let you load and unload licenses clusterwide.
- The SET ENVIRONMENT/NODE command now accepts logical names as values.

The following sections describe these features.

2.3.1 ALF Support Added to SYSMAN

You can use the SYSMAN Utility to perform specific functions of the auto-login facility (ALF). ALF lets you associate a terminal with a user name, which allows certain users to log in without specifying a user name. To use ALF commands, you need read and write access to the SYSALF database (SYS\$SYSTEM:SYSALF.DAT, by default).

SYSMAN supports the ALF commands ADD and REMOVE. These commands are described in the following sections.

2.3.1.1 ALF ADD Command

To add a new record to the ALF database, use the ALF ADD command in the following format:

```
SYSMAN> ALF ADD device username [/TERMINAL] [/PORT] [/PROXY] [/LOG]
```

where:

- *device* is the terminal or port name that you want to assign to a user name. Note that *device* must be a terminal name if you do not specify qualifiers on the command line.

- *username* is the user name of the account that you want to assign to a particular terminal or port.
- The /TERMINAL qualifier causes SYSMAN to treat *device* as a terminal name. This is the default behavior.
- The /PORT qualifier causes SYSMAN to treat *device* as a port name. If the port name contains a special character such as a “/”, or if it contains lowercase letters that you want to preserve, you must enclose the port name within quotation marks.
- The /PROXY qualifier causes SYSMAN to treat *device* as a port name. SYSMAN also checks that *device* is in the NODE::USERNAME format.
- The /LOG qualifier causes SYSMAN to echo the device name and the user name added to the ALF database.

For example, to assign terminal TTA3 to user FMARTIN, enter the following command:

```
SYSMAN> ALF ADD TTA3 FMARTIN
```

To assign port NJB34C3/LC-1-3 to user FMARTIN, enter the following command:

```
SYSMAN> ALF ADD "NJB34C3/LC-1-3" FMARTIN /PORT
```

2.3.1.2 ALF REMOVE Command

To remove one or more records from the ALF database, use the ALF REMOVE command in the following format:

```
SYSMAN> ALF REMOVE [device] [/USERNAME=username] [/CONFIRM] [/LOG]
```

where:

- *device* specifies the terminal or port name whose record you want to delete in ALF. You can use wildcard characters in the terminal or port name.

NOTE: When you specify *device* to remove a record from the ALF database, be sure to use the correct format. Include special characters such as underscores (_) and colons (:). Enter the ALF SHOW command to display the device name format.

- The /USERNAME qualifier allows you delete a record in ALF by specifying a username rather than a terminal or port name. You can use wildcard characters with the /USERNAME qualifier.
- The /CONFIRM qualifier causes SYSMAN to display a message asking if you really want to delete the record.
- The /LOG qualifier causes SYSMAN to echo the device name and user name to be removed from the ALF database.

2-8 System Management Features

SYSMAN Utility

For example, to remove the record for terminal TTA1 from the ALF database, enter the following command:

```
SYSMAN> ALF REMOVE _TTA1:
```

To remove all records for user name SMITHSON, enter the following command:

```
SYSMAN> ALF REMOVE/USERNAME=SMITHSON
```

You can use the /*USERNAME* qualifier and also specify a terminal or port name on the same command line. For example, to remove all records for a user name beginning with the letter *B* and all records for terminals named TTA*x*, enter the following command:

```
SYSMAN> ALF REMOVE _TTA*/USERNAME=B*
```

2.3.1.3 ALF SHOW Command

To display records in the ALF database, use the ALF SHOW command in the following format:

```
SYSMAN> ALF SHOW [device] [/USERNAME=username] [/OUTPUT=filespec]
```

where:

- *device* specifies the terminal or port name whose record you want to display. You can use wildcard characters in the terminal or port name.
- The /*USERNAME* qualifier lets you display the records held by the specified user name.
- The /*OUTPUT* qualifier lets you direct the output of the command to a file.

For example, the following command selects the records for all terminals named TTA*x* and for the user name MANESS, and directs a listing of the records to a file:

```
SYSMAN> ALF SHOW TTA* /USERNAME=MANESS /OUTPUT=ALF.TXT
```

2.3.2 Change in Privileges Granted on Remote Nodes

Before executing any command in an environment, SYSMAN verifies that you are an authorized user. Prior to Version 5.2, SYSMAN verified that you were an authorized user by checking the user authorization file on each node in the environment. If you were an authorized user, SYSMAN granted the privileges, rights, and defaults of the remote node's system user authorization file and rights database file.

For clusters that have common SYSUAF and RIGHTSLLIST databases, VMS Version 5.2 changes the way SYSMAN grants privileges and rights on remote nodes. Instead of requiring a lookup on each remote node, SYSMAN now grants the same privileges, rights, and default directory in effect on the local node, provided the following conditions exist:

- The node where the commands are being executed is part of the same cluster as the node where the commands are being entered.
- The device name and file identification for the SYSUAF file match on the two nodes.
- The device name and file identification for the RIGHTSLIST file match on the two nodes.

If these conditions are not met, SYSMAN grants only the privileges and rights in the remote system's SYSUAF and RIGHTSLIST files.

2.3.3 CONFIGURATION SHOW CLUSTER Command–Enhanced Display

For VMS Version 5.2, the multicast address of a cluster has been added to the display for the CONFIGURATION SHOW CLUSTER command. For example:

```
SYSMAN> CONFIGURATION SHOW CLUSTER
Node BLUES: Cluster group number: 45932
Multicast address: AB-00-04-01-F2-FF
```

2.3.4 LICENSE Commands

For VMS Version 5.2, the SYSMAN Utility lets you load and unload licenses on multiple systems and on systems that are not local. To use these commands, you need the following privileges: CMKRNL, SYSNAM, and SYSPRV.

The SYSMAN commands LICENSE LOAD and LICENSE UNLOAD are similar to the LICENSE LOAD and LICENSE UNLOAD commands of the License Management Facility (LMF). For more information about LMF, see the *VMS License Management Utility Manual*.

2.3.4.1 LICENSE LOAD Command

To activate licenses registered in the LICENSE database, use the LICENSE LOAD command in the following format:

```
SYSMAN> LICENSE LOAD product [/DATABASE=filespec] [/PRODUCER=string]
```

where:

- *product* is the name of the product whose license you want to activate.
- The /DATABASE qualifier lets you specify the location of the LICENSE database. SYS\$COMMON:[SYSEXE]LMF\$LICENSE.LDB is the default file specification for the database. You do not need to use the /DATABASE qualifier if you use the default LICENSE database name and location.
- The /PRODUCER qualifier lets you specify the name of the company that owns the product for which you have a license. Use this qualifier only if the product is from a company other than Digital.

2-10 System Management Features

SYSMAN Utility

For example, the following command activates the license for DEC FORTRAN:

```
SYSMAN> LICENSE LOAD FORTRAN
```

For more information about the LICENSE LOAD command, see the *VMS License Management Utility Manual*.

2.3.4.2 LICENSE UNLOAD Command

To deactivate licenses registered in the LICENSE database, use the LICENSE UNLOAD command in the following format:

```
SYSMAN> LICENSE UNLOAD [product] [/PRODUCER=string]
```

where:

- *product* is the name of the product whose license you want to deactivate. If you enter the LICENSE LOAD command without specifying a product name, the system deactivates all available registered licenses.
- The /PRODUCER qualifier allows you to specify the name of the company that owns the product for which you have a license. Use this qualifier only if the product is from a company other than Digital.

For example, the following command deactivates the license for DEC FORTRAN.

```
SYSMAN> LICENSE UNLOAD FORTRAN
```

For more information about the LICENSE UNLOAD command, see the *VMS License Management Utility Manual*.

2.3.5 SET ENVIRONMENT/NODE Command

The SET ENVIRONMENT/NODE command lets you specify which nodes subsequent SYSMAN commands will act upon. For VMS Version 5.2, the SET ENVIRONMENT/NODE command accepts logical names as values. Before you define logical names to use with SYSMAN, you must create the logical name table SYSMAN\$NODE_TABLE as follows:

```
$ CREATE/NAME_TABLE/PARENT=LNMS$SYSTEM_DIRECTORY SYSMAN$NODE_TABLE
```

You might want to include this command in SYS\$MANAGER:SYLOGICALS.COM, which is executed during system startup.

The following example demonstrates how you can define logical names to be a node or list of nodes, and use those logical names to establish the SYSMAN environment:

```
$ DEFINE FORTRAN ALICE,BILL,DIRK/TABLE=SYSMAN$NODE_TABLE
$ DEFINE PASCAL DIRK,JILL,CI_NODES/TABLE=SYSMAN$NODE_TABLE
$ DEFINE CI_NODES SYS2,SYS8/TABLE=SYSMAN$NODE_TABLE
$ RUN SYS$SYSTEM:SYSMAN
SYSMAN> SET ENVIRONMENT/NODE=(FORTRAN,PASCAL)
Remote Password:
```

```
%SYSMAN-I-ENV, current command environment:  
  Individual nodes: ALICE,BILL,DIRK,JILL,SYS2,SYS8  
  At least one node is not in the local cluster.  
  Username SYSTEM      will be used on nonlocal nodes.
```

This feature is useful when you want to organize the nodes in your VAXcluster environment according to specific categories (for example, all CI-based nodes or all nodes with FORTRAN installed).

2.4 License Management Facility

VMS Version 5.2 includes the following new LMF features:

- Changes to the LICENSE LIST command improve performance when viewing large LICENSE databases. This command provides the same data as the VMS Version 5.0 version, but offers an improved display.
- A new version of the LMF command procedure VMSLICENSE.COM that provides the following enhancements:
 - You can now use the prompt-based VMSLICENSE interface in place of the LICENSE LIST, LICENSE CANCEL, and LICENSE MODIFY commands.
 - Each time you enter data during a VMSLICENSE session, the command procedure retains the data to use as default data for the next PAK registration or license management task.
 - You can now specify the license management task (register, amend, modify, cancel, list) as a parameter on the DCL command line. Use this to bypass the first menu of the procedure.
 - You can now register PAKs and PAAM amendments using batch processing. You can enter the name of a data file as a parameter when you invoke VMSLICENSE.COM. Data files can contain data for one or more PAKs. VMSLICENSE remembers data entered previously and provides default data to save typing time.

For more information, see the *VMS License Management Utility Manual*.

2.5 NETCONFIG.COM Security Enhancements

In VMS Version 5.2, the DECnet network configuration command procedure NETCONFIG.COM has been enhanced to provide several options for restricting default access. A new command procedure for existing networked systems, NETCONFIG_UPDATE.COM, described in Section 2.6, has been created for the same purpose.

You can plan the appropriate level of default access for your system and implement that plan by responding to a few questions posed by NETCONFIG.COM. NETCONFIG.COM then automatically records your choices in the UAF (user authorization file) and in the network configuration database.

2-12 System Management Features

NETCONFIG.COM Security Enhancements

Previously, NETCONFIG.COM created one default account named DECNET. That account provided default access to all network objects and applications that were not restricted by other forms of access control (for example, proxy accounts and access control lists). If you chose to limit default access, it was necessary to manually enter all the appropriate commands in the UAF, using the Authorize Utility, and in the network configuration database, using NCP commands.

2.5.1 Default Access Options

NETCONFIG.COM provides two different ways to limit default access. The most restrictive form is to not create the default DECnet account but to grant default access for certain system objects by creating a default account for each one that you want to use. Using NETCONFIG.COM, you can create an account for one or more of the following network objects:

- MAIL
- File access listener (FAL)
- PHONE
- Network management listener (NML)
- Loopback mirror (MIRROR)
- VMS Performance Monitor (VPM)

The second, less restrictive form of default access is to create a default DECnet account but to disable default access to user-written programs and procedures (also known as TASK objects). Default access for system objects is still enabled.

You can still create an unrestricted default DECnet account that includes default access to TASK objects. This type of access is suitable for systems with low security requirements. To do so, you must override the defaults provided by NETCONFIG.COM.

NOTE: If you do not create the default DECnet account, you must create a default account for MAIL and VPM, if you want to use them. The same is true for the MIRROR object if you want to use the User Environment Test Package (UETP) to test the network connection.

FAL, if enabled by the default DECnet account or a separate default account, makes a system vulnerable to unauthorized access. Digital advises against creating a default account for FAL. Other, more secure access methods are available. For more information about more secure access methods and the security implications of a FAL account, refer to the *Guide to VMS System Security*.

The MIRROR object is used for loopback testing. To test your network connection with VAX UETP you must create a default account for the MIRROR object, if you did not create the default DECnet account.

The VPM object is used by the Monitor Utility in VAXcluster configurations to obtain performance information about VAXcluster members. If your system is a member of a VAXcluster and the cluster manager wants to use the Monitor Utility to collect such information, you must create a default account for the VPM object, if you did not create the default DECnet account.

2.5.2 Security Benefits

The DECnet account provides default access for all incoming links (unless this access is overridden by other forms of access control). However, default accounts for any of the system objects named in the NETCONFIG.COM procedure limit access to these objects. Default accounts for selected objects, when used with other system security facilities, enable a system or network manager to monitor these accounts and to detect unauthorized access.

For each default account that you create, NETCONFIG.COM generates a password and registers it in your network configuration database. Such system-generated passwords are more secure than the passwords that users typically create.

2.5.3 Questions Posed by NETCONFIG.COM

NETCONFIG.COM poses the following questions (the responses in brackets are the default values):

Do you want a default DECnet account? [NO]:

(The following question will be asked only if you said YES to a default DECnet account.)

Do you want default access to the TASK object disabled? [YES]:

(The following questions will be asked regardless of whether you said YES or NO to a default DECnet account.)

Do you want a default account for the MAIL object? [YES]:

Do you want a default account for the FAL object? [NO]:

Do you want a default account for the PHONE object? [YES]:

Do you want a default account for the NML object? [YES]:

(The following questions will be asked only if you said NO to a default DECnet account.)

Do you want a default account for the MIRROR object? [YES]:

Do you want a default account for the VPM object? [YES]:

2.6 New NETCONFIG_UPDATE.COM for Existing Networks

NETCONFIG_UPDATE.COM is a new command procedure for existing networks that provides the same security enhancements for default access that are provided by NETCONFIG.COM (see Section 2.5). It also provides a secondary procedure for modifying members of a VAXcluster. Both procedures are described in the following sections.

2.6.1 Benefits of NETCONFIG_UPDATE.COM

NETCONFIG_UPDATE.COM, unlike NETCONFIG.COM, configures default access only. It performs no other network configuration. Therefore, when you use NETCONFIG_UPDATE.COM to specify changes to default access, everything else in the configuration database remains unchanged.

NETCONFIG_UPDATE.COM, like NETCONFIG.COM, generates passwords for each account that you create for default access and for any existing default accounts that you decide to keep in your configuration database. For example, if you currently have a default account for MAIL and decide to keep it, NETCONFIG_UPDATE.COM generates a new password for it and replaces the existing password with the new one.

2.6.2 Using NETCONFIG_UPDATE.COM in a VAXcluster

NETCONFIG_UPDATE.COM provides a secondary procedure that updates the default access of VAXcluster members. After you run NETCONFIG_UPDATE.COM on one member of a VAXcluster, the procedure detects that it is a VAXcluster member and instructs you to run SYS\$COMMON:[SYSMGR]UPDATE_CLUSTER_MEMBERS.COM on the other VAXcluster members. This secondary procedure will modify the default access of each VAXcluster member exactly as you modified that of the first member.

With the SYSMAN Utility (see the *VMS SYSMAN Utility Manual*), you can use the SET ENVIRONMENT/CLUSTER command to execute this secondary procedure only once. The default access of all the remaining VAXcluster members will be updated automatically.

2.7 Disk and Tape Features

VMS Version 5.2 includes the following new disk and tape features:

- DSSI-based disk configurations. These configurations use the DSSI bus to connect as many as six RF30 disk drives between DSSI controllers on pairs of MicroVAX 3300/3400 CPUs. Like HSC-based configurations, DSSI-based configurations provide multihost disk access, flexibility, expansion potential, and online maintenance and backup capability. However, they do not support volume shadowing.

- Failover of dual-pathed disks is triggered by MSCP event messages. Disks need no longer be mounted locally on serving nodes.

2.8 Backup Utility

This section describes the following new Backup Utility (BACKUP) features:

- Performance enhancements that cause BACKUP save and copy operations to complete more quickly on systems that are configured correctly
- Faster cyclic redundancy checking (CRC) emulation for processors that emulate CRC in software, resulting in a significant performance enhancement for BACKUP on these processors
- Support for the control character Ctrl/T, which returns information about the online or standalone BACKUP operation in progress

2.8.1 Performance Enhancements

Version 5.2 of the Backup Utility includes a new method of scanning files on the input disk. This new file-scanning method results in faster save and copy operations on systems that are configured correctly. (It does not improve BACKUP's performance during restore, compare, verify, or list operations, however.) Prior to Version 5.2, disk head movement on the input disk constrained the speed at which BACKUP could save or copy files.

In order to take full advantage of the new BACKUP file-scanning method, you must change the values of certain user authorization file (UAF) and System Generation Utility (SYSGEN) parameters. Sections 2.8.1.1 and 2.8.1.2 specify which parameters you need to change.

2.8.1.1 Setting Up the BACKUP Account

BACKUP's new file-scanning method depends on the values of some user authorization file (UAF) parameters of the BACKUP account; the BACKUP account is the account from which you perform BACKUP operations. These UAF parameters define process quotas, which are the amounts of system resources available to a process created by the account. Digital recommends that you change the values of these UAF parameters for the BACKUP account. See the *VMS Authorize Utility Manual* for more information about modifying the values of UAF parameters.

Table 2-2 describes the UAF parameters that should be modified and supplies values that provide the maximum amount of resources to BACKUP. These values may not provide the best performance in all cases, however. They are intended to be general guidelines.

NOTE: BACKUP bases its memory consumption on the WSQUOTA value, not WSEXTENT.

2-16 System Management Features

Backup Utility

Table 2-2: UAF Process Quotas for the BACKUP Account

UAF Parameter	Meaning	Recommended Value
WSQUOTA	The number of pages of memory the working set of the process can consume.	Equal to SYSGEN parameter WSMAX
WSEXTENT	The absolute limit of physical memory allowed to the process.	Equal to WSQUOTA
PGFLQUO	The number of pages of memory your process is allowed in the page file.	Greater than or equal to WSEXTENT
FILLM	The number of files that can be open simultaneously. BACKUP scans this number of files at one time.	Equal to the SYSGEN parameter CHANNELCNT
DIOLM	The number of direct I/O operations (usually disk operations) that can be outstanding simultaneously.	Maximum of either (3 x FILLM) or 4,096
ASTLM	The number of asynchronous system traps that can be queued to the process simultaneously.	Maximum of either (3 x FILLM) or 4,096
BIOLM	The maximum number of buffered I/O operations that can be outstanding simultaneously.	Less than or equal to FILLM
BYTLM	The total number of bytes of memory that can be outstanding for buffered I/O operations.	Greater than or equal to the following value: (256 x FILLM) + (6 x DIOLM)
ENQLM	The maximum number of locks that can be queued simultaneously.	Greater than FILLM

Table 2-3 lists a set of UAF parameter values that may be useful for your configuration. You can choose to set the values for WSQUOTA and FILLM lower than these values under the following circumstances:

- If your disks are highly fragmented, lower values prevent BACKUP from becoming highly CPU-intensive.

- If you use BACKUP during periods of heavy system use, lower values prevent BACKUP from consuming too many system resources.

NOTE: If you decrease the values of UAF parameters other than WSQUOTA and FILLM, use the ratios in Table 2-2 to determine appropriate values.

Alternatively, you can choose to set the values higher than these suggested values if files are stored contiguously on your disks and if you perform BACKUP operations during periods of light system use.

Table 2-3: Suggested Values for UAF Process Quotas

UAF Parameter	Value
WSQUOTA	16,384
WSEXTENT	Greater than or equal to WSQUOTA
PGFLQUO	32,768
FILLM	128
DIOLM	4,096
ASTLM	4,096
BIOLM	128
BYTLM	65,536
ENQLM	256

After changing UAF parameters, log out of the BACKUP account and log back in, allowing the new values of the UAF parameters to be used.

2.8.1.2 Setting System Generation Utility (SYSGEN) Parameters

In order for the new BACKUP file-scanning method to work efficiently, the System Generation Utility (SYSGEN) parameters CHANNELCNT and WSMAX must be set to appropriate values. If the BACKUP account's FILLM value is greater than the value of the SYSGEN parameter CHANNELCNT, CHANNELCNT constrains the number of files that can be opened at any one time. If the BACKUP account's WSQUOTA value is greater than the value of the SYSGEN parameter WSMAX, WSMAX constrains the number of pages of memory that the working set of the process can consume. See the *VMS System Generation Utility Manual* for more information about changing the values of SYSGEN parameters.

After changing SYSGEN parameters, shut down and reboot the system, allowing the new values of the parameters to be used.

2.8.1.3 Understanding Why the Output Device Seems Idle

Because BACKUP can scan many files at a time, it is possible that no data will be sent to the output device for up to several minutes after the save or copy operation begins. This does not indicate that BACKUP is performing slowly or that your output device is not working correctly. Depending on the values of the UAF parameters and the SYSGEN parameters, BACKUP's new file-scanning method requires a certain amount of time to become established. When the file-scanning is completed, BACKUP sends the data to the output device more efficiently than it did before VMS Version 5.2.

2.8.1.4 /BUFFER_COUNT Command Qualifier Is Now Obsolete

The new file-scanning method used by BACKUP makes the command qualifier /BUFFER_COUNT obsolete. Previously, this command qualifier specified the number of buffers used in a save, compare, or restore operation to or from a tape. BACKUP now determines how many buffers to use, depending on the amount of memory available to the account performing the BACKUP operation and the number of files that account can open simultaneously.

You can still specify the /BUFFER_COUNT qualifier, however, although it has no effect. This ensures that command procedures written before VMS Version 5.2 will still operate correctly. Digital recommends that you remove the /BUFFER_COUNT qualifier from command procedures.

2.8.2 Cyclic Redundancy Checking Emulation Improvements

The method for performing cyclic redundancy checking (CRC) emulation is now approximately 40% faster than the method used before VMS Version 5.2. This is not a BACKUP-specific improvement, but it does improve BACKUP performance on processors that emulate CRC in software. BACKUP operations that use cyclic redundancy checking (CRC is applied by default) now require significantly less time to complete on the following processors, all of which emulate CRC in software:

- MicroVAX II/VAXstation II
- MicroVAX 2000/VAXstation 2000
- MicroVAX 3200/VAXstation 3200
- MicroVAX 3500/VAXstation 3500
- MicroVAX 3600
- VAX 6200

2.8.3 Pressing Ctrl/T to Obtain Information About BACKUP Operations

Version 5.2 of the VMS operating system supplies an additional two lines of information when you press Ctrl/T during an online or standalone BACKUP operation. Ctrl/T interrupts execution of the BACKUP command, and displays three lines of information. The first line displays information about the current process (node name, process name, system time, currently running image, elapsed CPU time, page faults, direct and buffered I/O operations, and pages in physical memory). The second line displays information about BACKUP input. The third line displays information about BACKUP output. For example, if you press Ctrl/T during a save operation, the second line displays the name of the last file scanned by BACKUP and the third line displays the save-set volume number, save-set block number, and the number of bytes in a block.

In order to use Ctrl/T, the command SET CONTROL=T must either appear in the system login command procedure or in your personal login command procedure. You can also enable Ctrl/T interactively by entering the DCL command SET CONTROL=T.

The following example shows what happens when you press Ctrl/T during a BACKUP save operation:

```
$ BACKUP/LOG DUA0:[MISHA]*.COM;* MUA0:COMPROCS.BCK/REWIND/LABEL=COMP
BACKUP-S-COPIED, copied DUA0:[MISHA]A.COM;32
BACKUP-S-COPIED, copied DUA0:[MISHA]B.COM;30
BACKUP-S-COPIED, copied DUA0:[MISHA]C.COM;16
[Ctrl/T]
SQUASH::MISHA 14:02:12 BACKUP CPU=00:00.18.44 PF=2101 IO=827 MEM=534
Last file scanned: DUA0:[NATASHA]D.DAT
Saveset volume: 1, saveset block: 35, (32256 byte blocks)
BACKUP-S-COPIED, copied DUA0:[MISHA]D.COM;2
BACKUP-S-COPIED, copied DUA0:[MISHA]E.COM;22
.
.
$
```

2.9 VMS DECwindows System Tailoring

Using DECW\$TAILOR, you can now tailor DECwindows systems. DECW\$TAILOR works much like VMSTAILOR; you can tailor DECwindows files “on” or “off.” Use the following command to run DECW\$TAILOR:

```
$ RUN SYS$UPDATE:DECW$TAILOR
```

The format and prompts are nearly identical to those displayed by VMSTAILOR. However, DECW\$TAILOR lets you tailor DECwindows files only.

Chapter 3

Programming Features

This chapter describes new programming features for VMS Version 5.2. These features include:

- Enhanced debugger
- New and modified system services
- Enhanced VMSINSTAL

3.1 Debugger

The Version 5.2 debugger provides the following new features:

- You can now debug multiprocess programs. Multiprocess programs are programs that run in more than one process. The following new and enhanced commands are used for multiprocess debugging:
 - CONNECT
 - DEFINE/PROCESS_GROUP
 - DISPLAY
 - DO
 - EXIT
 - QUIT
 - SET MODE [NO]INTERRUPT
 - SET PROCESS
 - SHOW PROCESS
 - SET PROMPT/[NO]SUFFIX
 - SET BREAK /ACTIVATING, /TERMINATING
 - CANCEL BREAK /ACTIVATING, /TERMINATING

3-2 Programming Features

Debugger

- SET TRACE /ACTIVATING, /TERMINATING
- CANCEL TRACE /ACTIVATING, /TERMINATING
- A new DECwindows version of the debugger provides a direct-manipulation style interface for workstations running VMS DECwindows.

In addition, enhancements were made to the following commands:

- The DISPLAY command now lets you create a display as well as modify an existing display.
- The SET SCOPE command now has a /CURRENT qualifier that lets you set the default scope search list relative to the routine call stack.

The Version 5.2 debugger lets you use Ctrl/C to interrupt program execution or abort a debugger command without interrupting the debugging session. The debugger prompt is displayed after you press Ctrl/C.

If your program already has a Ctrl/C AST routine enabled, the new SET ABORT_KEY command lets you reassign the abort function to another control-key sequence.

For more information about multiprocess debugging, the DECwindows debugger, and command enhancements, see the *VMS Debugger Manual*.

3.2 System Services

Version 5.2 includes the following new system services:

New Service	Function
\$DEVICE_SCAN	Scans across the system for devices
\$PROCESS_SCAN	Scans across the system or cluster for processes

For Version 5.2, processes are visible clusterwide. As a result, significant changes have been made to the following system services:

- \$CANWAK
- \$DELPRC
- \$FORCEX
- \$GETJPI
- \$RESUME
- \$SCHDWK
- \$SETPRI

- `$SUSPEND`
- `$WAKE`

Changes have also been made to the `$GETUAI`, `$SETUAI`, `$MOUNT`, and `$DISMOUNT` system services. For a complete description of the new and changed system services, see Chapter 4.

3.3 VMSINSTAL New Features

In order to simplify the task of creating the installation procedure for a layered product, VMSINSTAL provides routines called **callbacks**. In your installation procedure (KITINSTAL.COM), you invoke callbacks to perform basic functions such as moving files, creating directories, and deleting files. After the callbacks execute, control returns to your installation procedure.

Version 5.2 includes the following new VMSINSTAL callbacks:

- `CHECK_NETWORK`
- `CHECK_PRODUCT_VERSION`
- `COMPARE_IMAGE`
- `GET_IMAGE_ID`
- `UPDATE_IDENTIFIER`

Version 5.2 also includes changes to the following callbacks:

- `SET`
- `CHECK_NET_UTILIZATION`
- `CREATE_DIRECTORY`
- `ASK`

In addition to new and changed callbacks, Version 5.2 includes changes and additions to the VMSINSTAL options parameter.

The following sections describe new VMSINSTAL features in detail.

3.3.1 Check_Network Callback

Use the `CHECK_NETWORK` callback to determine whether the network is running. The installation can then either proceed or terminate based on the status of the network.

Note that VMSINSTAL no longer checks the network by default. If your product needs to check the network before beginning the installation, you must use this callback. Products should display messages to the installer that indicate whether the network should be running while performing the installation. Use the following command line format to invoke the `CHECK_NETWORK` callback:

3-4 Programming Features

VMSINSTAL New Features

```
$ VMI$CALLBACK CHECK_NETWORK symbol
```

Parameters on the command line indicate the following:

symbol

Use this parameter (P2) to specify a global symbol whose value reflects the status of the network. This symbol returns a value of true (1) if the network is running and false (0) if the network is not running.

Following is an example of a command line that invokes the CHECK_NETWORK callback:

```
$ VMI$CALLBACK CHECK_NETWORK NETSTAT
```

In the example, the CHECK_NETWORK callback defines the symbol *NETSTAT* as true when the network is running and false when the network is not running.

The CHECK_NETWORK callback always returns VMI\$_SUCCESS.

3.3.2 CHECK_PRODUCT_VERSION Callback

Use the CHECK_PRODUCT_VERSION callback to check the version of another product. VMSINSTAL extracts the version number from the file identification field of the image file associated with the product. VMSINSTAL then compares this version number to the minimum version specified in the command line.

You can use this callback if your product requires the existence of a specific version of another product.

Use the following command line format to invoke the CHECK_PRODUCT_VERSION callback:

```
$ VMI$CALLBACK CHECK_PRODUCT_VERSION symbol file-spec minimum_version [option]
```

Parameters on the command line indicate the following:

symbol

Use this parameter (P2) to specify a symbol that returns a value of true (1) when the product meets the minimum version requirement specified in P4. When the product does not meet the minimum version requirement, this symbol returns a value of false (0).

file-spec

Use this parameter (P3) to indicate the full file specification of the prerequisite product's image file. This is the file from which VMSINSTAL extracts the version number.

minimum_version

Use this parameter (P4) to specify the minimum version required to install your product. Specify the minimum version using the following format:

```
tvv.u-m
```

where:

t is the type of release (for example V for released version, T for field test version)
 vv is the major version number
 u is the update number
 m is the maintenance number

For example, you might specify the following minimum version: V3.0-5.

option

Use this parameter (P5) to specify the name of the product for which you are performing the version check. If the product does not meet the minimum version requirement specified in P4, VMSINSTAL displays the following message to the installer:

```
This kit requires at least: product_name minimum_version
Please install a proper version of product_name
before installing this product.
```

Enclose the product name with quotation marks.

Following is an example of a command line that invokes the CHECK_PRODUCT_VERSION callback:

```
$ VMI$CALLBACK CHECK_PRODUCT_VERSION MY$CHECK VMI$ROOT:[SYSEXE]RDO.EXE V3.0
```

In the example, VMSINSTAL determines whether the RDO image is version 3.0 or higher; if the image is version 3.0 or higher, VMSINSTAL defines symbol MY\$CHECK as true.

This callback always returns VMI\$_SUCCESS.

3.3.3 COMPARE_IMAGE Callback

The COMPARE_IMAGE callback compares the file identification field of two image files and returns a value that indicates which file is a more recent version.

Use the following command line format to invoke the COMPARE_IMAGE callback:

```
$ VMI$CALLBACK COMPARE_IMAGE symbol file-spec1 file-spec2
```

Parameters on the command line indicate the following:

symbol

Use this parameter (P2) to specify a symbol that returns one of the following values:

- VMI\$K_KIT_VER_OLDER—Indicates that the image file specified in P4 is an older version than the image file specified in P3
- VMI\$K_KIT_VER_NEWER—Indicates that the image file specified in P4 is a newer version than the image file specified in P3

3-6 Programming Features

VMSINSTAL New Features

- **VMI\$K_KIT_VER_SAME**—Indicates that the image files specified in P3 and P4 are the same version
- **VMI\$K_KIT_VER_NOMATCH**—Indicates that the facility names of the files specified in P3 and P4 do not match

file_spec1

Use this parameter to indicate the full file specification of one of the image files.

file-spec2

Use this parameter to indicate the full file specification of the other image file.

Following is an example of a command line that invokes the **COMPARE_IMAGE** callback:

```
$ VMI$CALLBACK COMPARE_IMAGE MY$CHECK VMI$ROOT:[SYSEXE]RDO.EXE VMI$KWD:RDO.EXE
```

The example compares two RDO images to determine which is the newest version and defines symbol **MY\$CHECK** accordingly.

This callback returns **VMI\$_SUCCESS** if the file specified in P4 is found. Otherwise, it returns **VMI\$_FAILURE**.

3.3.4 GET_IMAGE_ID Callback

The **GET_IMAGE_ID** callback extracts the file identification field data for a file.

Use the following command line format to invoke the **GET_IMAGE_ID** callback:

```
$VMI$CALLBACK GET_IMAGE_ID symbol file-spec
```

Parameters on the command line indicate the following:

symbol

Use this parameter (P2) to specify a global symbol that **VMSINSTAL** equates to the image file identification string from the image header section.

file-spec

Use this parameter (P3) to specify the full file specification of the file for which you want to obtain the file identification string.

Following is an example of a command line that invokes the **GET_IMAGE_ID** callback:

```
$ VMI$CALLBACK GET_IMAGE_ID MY$IMAGE_ID VMI$ROOT:[SYSEXE]RDO.EXE
```

This callback returns **VMI\$_SUCCESS** if the file is found; if the file is not found, it returns **VMI\$_FAILURE**.

3.3.5 UPDATE_IDENTIFIER Callback

The UPDATE_IDENTIFIER callback modifies an identifier in the rights database.

Use the following command line format to invoke the UPDATE_IDENTIFIER callback:

```
$ VMI$CALLBACK UPDATE_IDENTIFIER id-name qualifiers
```

Parameters on the command line indicate the following:

id-name

Use this parameter (P2) to specify the name of the identifier to be modified.

qualifiers

Use this parameter (P3) to specify qualifiers for the UPDATE_IDENTIFIER callback. You can specify the same qualifiers that are available for the MODIFY/IDENTIFIER command of the Authorize Utility. For more information about the MODIFY/IDENTIFIER command, see the *VMS Authorize Utility Manual*.

Following is an example of a command line that invokes the UPDATE_IDENTIFIER callback:

```
$ VMI$CALLBACK UPDATE_IDENTIFIER ACCOUNTING "/VALUE=UIC:[300,21]"
```

In this example, the UPDATE_IDENTIFIER callback specifies a new value for the ACCOUNTING identifier.

This callback returns VMI\$_SUCCESS if the identifier is successfully modified. The callback returns VMI\$_FAILURE if the identifier is not successfully modified.

3.3.6 SET Callback—New ASK_CASE Option

Use the new ASK_CASE option to the SET callback to specify the default case (for example, uppercase or lowercase) in which input from the installer is returned to the installation procedure.

You can override the default case for a specific ASK callback by specifying the appropriate ASK option (U, L, or M).

Use the following command line format to invoke the SET ASK_CASE option:

```
$ VMI$CALLBACK SET ASK_CASE case
```

The parameter on the command line indicates the following:

case

Use this parameter (P3) to specify the default case. You can specify the following values for case:

- UPPER—When you specify UPPER, all input is returned in uppercase. By default, input from the installer is returned in uppercase.
- LOWER—When you specify LOWER, all input is returned in lowercase.

3-8 Programming Features

VMSINSTAL New Features

- **MAINTAIN**—When you specify **MAINTAIN**, all input is returned in the same case as that entered by the installer.

The **SET ASK_CASE** option always returns **VMI\$_SUCCESS**.

3.3.7 SET Callback—New POSTINSTALL Option

Use the new **POSTINSTALL** option to the **SET** callback when you want a product's **KITINSTAL.COM** procedure to be called after all files have been moved to their target directories. This callback is useful when you must have all new files in place before completing an installation.

This callback passes the string **VMI\$_POSTINSTALL** to the **KITINSTAL.COM** procedure in **P1**. The **KITINSTAL.COM** procedure is executed immediately before the **IVP** phase.

Use the following command line format to invoke the **SET POSTINSTALL** option:

```
$ VMI$CALLBACK SET POSTINSTALL
```

The **SET POSTINSTALL** option always returns **VMI\$_SUCCESS**.

3.3.8 CHECK_NET_UTILIZATION Callback—New Parameters

Using the new **VMSINSTAL** installation option, **AWD** (alternate working device), an installer can now specify an alternate device for the **VMSINSTAL** working directory. As a result, the **CHECK_NET_UTILIZATION** callback has been modified to let you specify peak space requirements for the product target device in **P4** and the alternate working device in **P5**. These parameters are optional.

For more information about the **AWD** option, see Section 3.3.11.2.

Use the following command line to invoke the **CHECK_NET_UTILIZATION** callback:

```
$ VMI$CALLBACK CHECK_NET_UTILIZATION symbol blocks [trg-blocks] [awd-blocks]
```

Parameters on the command line indicate the following:

symbol

Use this parameter (**P2**) to specify a global symbol that the callback uses to indicate whether the disk has sufficient net free space to install your product. The callback returns the value **true (1)** for this symbol if there are sufficient net free blocks available; otherwise, its value is **false (0)**.

blocks

Use this parameter (**P3**) to specify the number of net free blocks required for your product, that is, the total number of blocks required to perform the installation (including the blocks required for the temporary working directory).

trg-blocks

Use this parameter (P4) to specify the number of free blocks that must be available on the target device (VMI\$ROOT) in order to accommodate the completed installation. This number does not include the number of blocks required for the temporary working area; this number is specified in P5. The values specified in P4 and P5 are used only when the product is installed using the AWD option.

awd-blocks

Use this parameter (P5) to specify the number of free blocks that must be available on the device associated with the alternate working device.

Following is an example of the command line used to invoke the CHECK_NET_UTILIZATION callback:

```
$ VMI$CALLBACK CHECK_NET_UTILIZATION CHECK$ 20000 12000 10000
```

The callback can function in the following ways, depending on whether or not the installer specifies the AWD option:

- If the installer does not specify the AWD option, the callback returns the value true (1) in the symbol CHECK\$ when the system has at least 20,000 free blocks on the VMI\$ROOT device. If the system does not have at least 20,000 free blocks, the callback returns a value of false (0).
- If the installer specifies the AWD option, the callback returns the value 1 (true) when the VMI\$ROOT device has at least 12,000 blocks and the alternate working device has at least 10,000 blocks. If either of these devices has less than the specified number of free blocks, the callback returns a value of 0 for the CHECK\$ symbol.

This callback always returns VMI\$_SUCCESS.

3.3.9 CREATE_DIRECTORY Callback—New Option

The CREATE_DIRECTORY callback now lets you create a system directory in the common area (VMS\$COMMON) without also creating the directory in the system-specific root of the installing system. To create a system directory in the common area only, use the following command line format:

```
$ VMI$CALLBACK CREATE_DIRECTORY COMMON name [qualifiers]
```

Parameters on the command line indicate the following:

COMMON

When you enter the keyword COMMON in this parameter (P2), the product directory is created under the system common area only.

3-10 Programming Features

VMSINSTAL New Features

name

Use this parameter to specify the name of the directory. The callback recognizes the keyword `COMMON` and makes the directory a rooted directory. Therefore, do not specify the system device designation and do not include the brackets usually used in directory name specifications.

qualifiers

Use this parameter (P4) to specify one or more of the `CREATE/DIRECTORY` command qualifiers: `/OWNER_UIC`, `/PROTECTION`, `/VERSION_LIMIT`. The entire parameter must be specified as a character string enclosed by quotation marks; for example, you might specify `"/OWNER_UIC=[SMITH]"`.

The command line in the following example creates the directory `[VMS$COMMON.MYPROD.LOGS]`:

```
$ VMI$CALLBACK CREATE_DIRECTORY COMMON MYPROD.LOGS
```

This callback always returns `VMI$_SUCCESS`.

3.3.10 ASK Callback—New Options

The ASK callback provides a method for obtaining information from the installer. This callback now provides three new options that control the case (for example, uppercase or lowercase) of input returned by the installer. These new options are `U` (uppercase), `L` (lowercase), and `M` (maintain input case).

Use the following command line format to invoke the ASK callback:

```
$ VMI$CALLBACK ASK symbol prompt [default_response] [options] [help]
```

To return input from the installer in uppercase, specify option `U`. To return input in lowercase, specify option `L`. To maintain the case as entered by the installer, specify `M`.

By default, input is returned from the installer in uppercase or the case specified by the `SET ASK_CASE` callback.

For more information about the ASK callback, see the *VMS Developer's Guide to VMSINSTAL*.

3.3.11 VMSINSTAL Command Line—New and Changed Options

`VMSINSTAL` provides the following new command line options:

- Restore Save Set and Pause option (RSP)
- Alternate Working Device option (AWD)

`VMSINSTAL` also has changes to the following command line options:

- Kit Debug (K)
- Release Notes (N)

VMSINSTAL command line options control the manner in which an installation is performed. You enter the VMSINSTAL command line using the following format:

```
@SYS$UPDATE:VMSINSTAL product_list source: [OPTIONS] [option-list][alternate_dir] [qualifiers]
```

The following sections describe the new and changed command line options in detail. For more information about VMSINSTAL options, see the installation guide for your processor and the *VMS Developer's Guide to VMSINSTAL*.

3.3.11.1 Restore Save Set and Pause (RSP) Option

The restore save set and pause (RSP) option is a debugging option that causes the installation procedure to pause after restoring each save set. In order to resume, you press Return at the prompt.

This option lets you test a kit without rebuilding the kit after each minor fix. For example, during the pause, you might edit a file that was previously restored or replace one version of a file with another.

You can also pause the installation procedure after a particular save set is restored by specifying the option in the following format:

```
RSP=s
```

where *s* indicates the save set to be restored before pausing.

For example, to pause after the installation procedure restores save set C, you specify the option as follows:

```
RSP=C
```

If you do not specify a save set with the RSP option, the installation procedure pauses after each save set is restored.

3.3.11.2 Alternate Working Device (AWD) Option

The alternate working device option lets you specify an alternate working device for the temporary working directory (defined as the logical VMI\$KWD). If you do not specify this option, VMSINSTAL creates the temporary working directory in the following location:

```
SYS$SPECIFIC:[SYSUPD.facvvu]
```

where *facvvu* indicates the product identification string.

Specify this option using the following format:

```
AWD=dev:[dir]
```

Parameters on the command line indicate the following:

3-12 Programming Features

VMSINSTAL New Features

dev:

Specifies the alternate working device.

dir

Specifies the directory under which the facvvu subdirectory will be created. Specifying a directory is optional. If you do not specify a directory, VMSINSTAL creates the working directory on the specified device with the following directory specification: [000000.facvvu]. If you specify a directory, VMSINSTAL creates the working directory as a subdirectory to the directory that you specify (for example, [WORK.facvvu]).

For example, to create the working directory [INSTALL.facvvu] on the alternate device DUA2, specify the following:

```
AWD=DUA2:[INSTALL]
```

3.3.11.3 Kit Debug (K) Option

The kit debug option now disables verification (SET NOVERIFY) while VMSINSTAL processes callbacks. Verification is reactivated after callbacks are processed.

3.3.11.4 Release Notes (N) Option

When an installer specifies the release notes option, VMSINSTAL displays the following enhanced menu:

```
Release notes included with this kit are always copied to SYS$HELP.
```

```
Additional Release Notes Options:
```

1. Display release notes
2. Print release notes
3. Both 1 and 2
4. None of the above

```
*Select option [2]:
```

```
*Queue name [SYS$PRINT]:
```

```
*Do you want to continue the installation [N]:
```

3.3.11.5 Specifying Multiple Options

Prior to Version 5.2, if you entered multiple options on the command line, you had to specify the options with no commas or spaces between letters. For Version 5.2, you can use commas to separate options in an options list. If you specify one of the new multicharacter options (RSP or AWD), you must separate the options with commas.

For example, to choose the kit debug (K) option and the restore save set and pause (RSP) option, you specify the following:

```
K,RSP
```

If the installer does not enter options on the command line, VMSINSTAL prompts for this information.

Chapter 4

System Services Features

VMS Version 5.2 includes the following new system services:

New Service	Function
<code>\$DEVICE_SCAN</code>	Scan across the system for devices
<code>\$PROCESS_SCAN</code>	Scan across the system or cluster for processes

The Device Scan system service, which is described in Section 4.10, enables you to find the names of all devices that match a specified set of search criteria. `$DEVICE_SCAN` can be used to produce a list of all disks, printers, or terminals on the local node. After `$DEVICE_SCAN` has located device information, you can use `$GETDVI` to further select the information, for instance, to find the names of all mounted disks or all terminals running at the same baud rate.

The Process Scan system service, which is described in Section 4.10, enables you to scan for processes across the cluster. For VMS Version 5.2, any VMS process can now be seen or modified from any node in a VAXcluster environment. Any system service that examines or modifies a process is now capable of examining or controlling a process that is located on a different node in the VAXcluster environment.

VMS Version 5.2 also includes changes to existing system services, as described in the following sections.

4.1 Modifications to `$SETUAI` and `$GETUAI`

VMS Version 5.2 includes the following changes to the `$SETUAI` and `$GETUAI` system services:

- New `$SETUAI` item codes

- `UAI$_PASSWORD`
 - `UAI$_PASSWORD2`
 - `UAI$_USER_DATA`

4-2 System Services Features

Modifications to \$SETUAI and \$GETUAI

- New \$SETUAI authorization flags
 - UAI\$V_DISIMAGE
 - UAI\$V_RESTRICTED
- New \$GETUAI item code
 - UAI\$_USER_DATA
- New \$GETUAI authorization flags
 - UAI\$V_DISIMAGE
 - UAI\$V_RESTRICTED

4.1.1 New Item Codes for \$SETUAI and \$GETUAI

The following new item codes have been added for the \$SETUAI system service:

New Item Code	Description
UAI\$_PASSWORD	When you specify UAI\$_PASSWORD, \$SETUAI sets the specified plaintext string as the primary password of the user and updates the password change date.
UAI\$_PASSWORD2	When you specify UAI\$_PASSWORD2, \$SETUAI sets the specified plaintext string as the secondary password of the user and updates the password change date.
UAI\$_USER_DATA	When you specify UAI\$_USER_DATA, \$SETUAI sets up to 255 bytes of information in the user data area of the system user authorization file (SYSUAF). This is the supported method for modifying the user data area of the SYSUAF. Digital no longer supports direct user modification of the SYSUAF. To clear all information in the user data area of the SYSUAF, specify \$SETUAI with a buffer length of zero.

The SYSPRV privilege is required to set any passwords (including the password of the calling process) or to modify the user data with \$SETUAI.

The UAI\$_PASSWORD and UAI\$_PASSWORD2 item codes provide the building blocks for designing a site-specific SET PASSWORD utility. If you create such a utility, you should set the LOCKPWD bit in the user authorization file (UAF) to prevent users from using the SET PASSWORD command and to prevent the LOGINOUT process from forcing password changes. If you create a site-specific SET PASSWORD utility, install the utility with SYSPRV privilege.

When specifying a password with UAI\$_PASSWORD or UAI\$_PASSWORD2, adhere to the following guidelines:

- The password must meet the minimum password length defined on the system.

- The password cannot exceed 32 characters in length.
- The password must be different from the previous password.

To clear the primary or secondary password, specify UAI\$_PASSWORD or UAI\$_PASSWORD2 with a buffer length of zero.

VMS Version 5.2 includes the following new item code for the \$GETUAI system service:

New Item Code	Description
UAI\$_USER_DATA	When you specify UAI\$_USER_DATA, \$GETUAI reads up to 255 bytes of information from the user data area of the system user authorization file (SYSUAF). You can read information written to the user data area from previous versions of the VMS operating system as long as the information adheres to the guidelines described in the <i>Guide to VMS System Security</i> .

4.1.2 New Authorization Flags for \$SETUAI and \$GETUAI

Two new authorization flags, UAI\$V_DISIMAGE and UAI\$V_RESTRICTED, are used in the creation of captive and restricted user accounts. (See the *Guide to VMS System Security* for a complete description of these flags.) Use the \$SETUAI system service to set the flags for the specified user. Use the \$GETUAI system service to determine whether the specified flag is set. The new flags are represented as bits in the UAI\$_FLAGS item code with the following symbolic names:

New Authorization Flags	Description
UAI\$V_DISIMAGE	When you specify UAI\$V_DISIMAGE, the user cannot issue the RUN or MCR commands or use the foreign command mechanism in DCL.
UAI\$V_RESTRICTED	Set the RESTRICTED flag (UAI\$V_RESTRICTED) to return the account to the level of security previously specified by the CAPTIVE authorization flag in earlier versions of the VMS operating system. (Under VMS Version 5.2, the security of accounts with the CAPTIVE flag set (UAI\$_V_CAPTIVE) has been increased by disallowing any access to the DCL command level and disallowing use of the INQUIRE verb in captive command procedures.)

4-4 System Services Features

Modifications to \$MOUNT

4.2 Modifications to \$MOUNT

The following flags have been added to the \$MOUNT system service:

New Flag	Description
MNT\$M_NOLABEL	The volume is to be mounted as a foreign volume; a foreign volume is not Files-11 structured. If you specify MNT\$M_NOLABEL, the following item codes may each appear in the item list only once: MNT\$_DEVNAM, MNT\$_VOLNAM, and MNT\$_LOGNAM. To specify MNT\$M_NOLABEL, the caller must either own the volume or have VOLPRO privilege.
MNT\$M_NOREBUILD	The volume to be mounted should be returned to active use immediately, without performing a rebuild operation. If a disk volume is improperly dismounted (such as during a system failure), you must rebuild it to recover any caching limits that were enabled on the volume at the time of the dismount. By default, MOUNT attempts the rebuild. For a successful rebuild operation that includes reclaiming all the available free space, you must mount all of the volume set members. Since the rebuild operation may take a significant amount of time, specifying MNT\$M_NOREBUILD is recommended. The volume should be rebuilt at a convenient time using the DCL SET VOLUME/REBUILD command to recover the free space.
MNT\$M_NOUNLOAD	The volume to be mounted is not to be unloaded when it is dismounted. Specifying MNT\$M_NOUNLOAD causes the volume to remain loaded when it is dismounted unless the dismount explicitly requests that the volume be unloaded.

4.3 Modifications to \$DISMOUNT

VMS Version 5.2 includes the following changes to the \$DISMOUNT system service:

New Flag	Description
DMT\$M_NOUNLOAD	Specifies that the volume is not to be physically unloaded after the dismount. If both the DMT\$M_UNLOAD and DMT\$M_NOUNLOAD flags are specified, the DMT\$M_NOUNLOAD flag is ignored. If neither flag is specified, the volume is physically unloaded, unless the DMT\$M_NOUNLOAD flag was specified on the \$MOUNT system service or the /NOUNLOAD qualifier was specified on the MOUNT command when the volume was mounted.

New Flag	Description
DMT\$M_OVR_CHECKS	Specifies that the volume should be dismounted without checking for open files, spooled devices, installed images, or installed swap and page files.
DMT\$M_UNIT	The specified device, rather than the entire volume set, is dismounted.
DMT\$M_UNLOAD	Specifies that the volume is to be physically unloaded after the dismount. If both the DMT\$M_UNLOAD and DMT\$M_NOUNLOAD flags are specified, the DMT\$M_NOUNLOAD flag is ignored. If neither flag is specified, the volume is physically unloaded, unless the DMT\$M_NOUNLOAD flag was specified on the \$MOUNT system service or the /NOUNLOAD qualifier was specified on the MOUNT command when the volume was mounted.

4.4 Modifications to Existing System Services for Clusterwide Process Accessibility

The following system services have been modified because a VMS process is now visible clusterwide:

Modified Service	Function
\$CANWAK	Cancel Wakeup
\$DELPRC	Delete Process
\$FORCEX	Force Image Exit
\$GETJPI	Get Job/Process Information
\$RESUME	Resume Process
\$SCHDWK	Schedule Wakeup for Process
\$SETPRI	Set Priority
\$SUSPEND	Suspend Process
\$WAKE	Wake Process

The descriptions of the **pidadr** and **prcnam** arguments have been changed for these system services. The **pidadr** argument can now refer to a process running on another node in the cluster. The process name can now specify a node name as well as the process name. This full process name can contain up to 23 characters.

4-6 System Services Features

Modifications to Existing System Services for Clusterwide Process Accessibility

In addition, these services now return the following status codes:

Status	Explanation
SS\$_INCOMPAT	The remote node is running a version of VMS prior to Version 5.2 and is unable to handle the request.
SS\$_NOSUCHNODE	The specified node is not currently a member of the cluster.
SS\$_REMRSRC	The remote node has insufficient resources to respond to the request. (Bring this error to the attention of your system manager.)
SS\$_UNREACHABLE	The remote node is a member of the cluster but is not accepting requests. (This is normal for a brief period early in the system boot process.)

4.5 Process Information Services

The VMS process information services enable you to gather information about processes. You can obtain information about one process or a group of processes on the local system or on remote nodes in a VAXcluster system. DCL commands such as SHOW SYSTEM and SHOW PROCESS use the process information services to display information about processes. You can use these services within your programs.

The following are process information system services:

- Get Job/Process Information (\$GETJPI)
- Process Scan (\$PROCESS_SCAN)

For detailed information about \$GETJPI, see the *VMS System Services Reference Manual*. For detailed information about \$PROCESS_SCAN, see Section 4.10.

4.6 Overview of \$GETJPI and \$GETJPI with \$PROCESS_SCAN

\$GETJPI was previously included with the process control system services. However, because \$GETJPI is used to obtain information about processes rather than control processes, \$GETJPI is now included with the process information services.

\$GETJPI returns information about processes. \$GETJPI uses the PID or the process name to obtain information about one process and the -1 wildcard to obtain information about all processes. \$GETJPI cannot perform a selective search—it can only search for one process in the cluster or for all processes on the local system. If you want to perform a selective search for information or get information about processes across the cluster, use \$GETJPI with \$PROCESS_SCAN.

Overview of \$GETJPI and \$GETJPI with \$PROCESS_SCAN

\$PROCESS_SCAN provides a process context that is used by \$GETJPI to return information about processes on the local system or across the cluster. \$PROCESS_SCAN can be used only with \$GETJPI; it cannot be used alone. The process context generated by \$PROCESS_SCAN is used by \$GETJPI like the -1 wildcard, except that it is initialized by calling the \$PROCESS_SCAN service instead of by a simple assignment statement. However, the \$PROCESS_SCAN context is more powerful and more flexible than the -1 wildcard. \$PROCESS_SCAN uses an item list to specify selection criteria to be used in a search for processes and produces a context longword that describes a selective search for \$GETJPI.

Using \$GETJPI with \$PROCESS_SCAN to perform a selective search is a more efficient way to locate information because information is returned only about the processes you have selected. For example, you can specify a search for processes owned by one user name, and \$GETJPI returns only the processes that match the specified user name. You can specify a search for all batch processes and \$GETJPI returns information only about processes running as batch jobs. You can specify a search for all batch processes owned by one user name and \$GETJPI returns information only about processes owned by that user name that are running as batch jobs.

4.6.1 Using the Process ID to Obtain Information

\$GETJPI returns information about processes by using the process identification (PID) or the process name. The PID is a 32-bit number that is unique for each process in the cluster. Specify the PID by using the **pidadr** argument. All the significant digits of a PID must be specified; only leading zeros can be omitted.

It might be preferable to use the **pidadr** argument instead of the **prcnam** argument when specifying a process to \$GETJPI, for the following reasons:

- The **pidadr** argument may be used to identify any process in the system, whereas the **prcnam** argument can be used only to identify processes that have the same UIC group number as the caller of \$GETJPI.
- \$GETJPI executes faster when you use **pidadr** rather than **prcnam**. When you specify **prcnam**, \$GETJPI must search a table of process names and UICs for an entry that contains the specified process name and the UIC group number of the calling process; this search is unnecessary when you use **pidadr**.

Table 4-1 shows how \$GETJPI operates given various values for the **prcnam** and **pidadr** arguments.

4-8 System Services Features

Overview of \$GETJPI and \$GETJPI with \$PROCESS_SCAN

Table 4-1: Process Identification

Process Name Specified?	Process ID Address Specified?	Contents of Process ID	Resultant Action by Services
No	No	–	The process identification of the calling process is used, but is not returned.
No	Yes	0	The process identification of the calling process is used and returned.
No	Yes	Process ID	The process identification is used and returned.
Yes	No	–	The process name is used. The process identification is not returned.
Yes	Yes	0	The process name is used and the process identification is returned.
Yes	Yes	Process ID	The process identification is used and returned. The process name is ignored.

4.6.2 Using the Process Name to Obtain Information

To obtain information about a process using the process name, specify the **prcnam** argument. Although a PID is unique for each process in the cluster, a process name is unique (within a UIC group) only for each process on a node. To locate information about processes on the local node, specify a process name string of 1- to 15-characters. To locate information about a process on a particular node, specify the full process name, which can be up to 23 characters long. The full process name is configured in the following way:

- 1 to 6 characters for the node name
- 2 characters for the colons (::) that follow the node name
- 1 to 15 characters for the local process name

Note that a local process name can look like a remote process name. Therefore, if you specify ATHENS::SMITH, the system checks for a process named ATHENS::SMITH on the local node before checking node ATHENS for a process named SMITH.

See the *VMS System Services Reference Manual* for more information about \$GETJPI. See the *Introduction to VMS System Services* for more information about process identification.

4.6.3 Modifications to \$GETJPI

You can still use \$GETJPI as you did before VMS Version 5.2. However, the \$GETJPI system service has been modified to work with \$PROCESS_SCAN. If you use \$PROCESS_SCAN with \$GETJPI, the process context (**pidctx**) is used by \$GETJPI as the **pidadr**.

\$GETJPI has also been modified to include new status codes and new item codes. The new status codes are described in Section 4.4. The new item codes are described in the following table:

Item Code	Function
JPI\$CPU_ID	ID of the CPU on which the process is running or on which it last ran, returned as -1 if the system is not a multiprocessor.
JPI\$GETJPI_CONTROL_FLAGS	Flags for options that control what actions \$GETJPI takes to retrieve the information (described in detail in the <i>Introduction to VMS System Services</i>).
JPI\$NODENAME	Name of the VAXcluster node on which the process is running.
JPI\$NODE_CSID	Cluster ID of the VAXcluster node on which the process is running.
JPI\$NODE_VERSION	VMS version number of the VAXcluster node on which the process is running.
JPI\$STS2	Second longword of process status flags.
JPI\$TERMINAL	Now returns up to eight bytes of information, including the colon (:) after the device name.
JPI\$TT_ACCPORNAM	Access port name for the terminal associated with the process. (The terminal name is returned by JPI\$TERMINAL.) If the terminal is on a terminal server, this item returns the terminal server name and the name of the line port on the server. If the terminal is a DECnet remote terminal, this item returns the source system node name and the user name on the source system. Otherwise, it returns a null string.
JPI\$TT_PHYDEVNAM	Physical device name of the terminal associated with the process. This name is the same as JPI\$TERMINAL unless virtual terminals are enabled, in which case JPI\$TERMINAL returns the name of the virtual terminal and JPI\$TT_PHYDEVNAM returns the name of the physical terminal. If JPI\$TERMINAL is null, or if the virtual terminal is disconnected from the physical terminal, JPI\$TT_PHYDEVNAM returns a null string.

4.7 Using \$GETJPI Alone

Using \$GETJPI without \$PROCESS_SCAN limits you to obtaining information about one process at a time or information about all processes on the local system. To obtain information about one process (either a local or a remote process), specify the PID or the process name. To obtain information about all processes on

4-10 System Services Features

Using \$GETJPI Alone

the local system, use the -1 wildcard as the **pidadr**. If no PID or process name is specified, \$GETJPI returns information about the calling process.

4.7.1 Requesting Information About a Single Process

Example 4-1 is a FORTRAN program that displays the process name and the PID of the calling program.

Example 4-1: Using \$GETJPI to Obtain Information About the Calling Process

! No process name or PID is specified; \$GETJPI returns data on the
! calling process.

```
PROGRAM CALLING_PROCESS

IMPLICIT NONE                                ! Implicit none

INCLUDE '($jptidef) /nolist'                ! Definitions for $GETJPI
INCLUDE '($ssdef) /nolist'                  ! System status codes

STRUCTURE /JPIITMLST/                        ! Structure declaration for
UNION                                        ! $GETJPI item lists
MAP
  INTEGER*2 BUFLen,
  2        CODE
  INTEGER*4 BUFADR,
  2        RETLENADR
END MAP
MAP                                          ! A longword of 0 terminates
  INTEGER*4 END_LIST                        ! an item list
END MAP
END UNION
END STRUCTURE
RECORD /JPIITMLST/                          ! Declare the item list for
2        JPILIST(3)                        ! $GETJPI

INTEGER*4 SYS$GETJPIW                        ! System service entry points

INTEGER*4 STATUS,                          ! Status variable
2        PID                               ! PID from $GETJPI

INTEGER*2 IOB(4)                            ! I/O status block for $GETJPI

CHARACTER*16
2        PRCNAM                            ! Process name from $GETJPI
INTEGER*2 PRCNAM_LEN                       ! Process name length
! Initialize $GETJPI item list
```

(continued on next page)

Example 4-1 (Cont.): Using \$GETJPI to Obtain Information About the Calling Process

```

JPILIST(1).BUFLEN      = 4
JPILIST(1).CODE       = JPI$_PID
JPILIST(1).BUFADR     = %LOC(PID)
JPILIST(1).RETLENADR  = 0
JPILIST(2).BUFLEN     = LEN(PCRNAM)
JPILIST(2).CODE       = JPI$_PCRNAM
JPILIST(2).BUFADR     = %LOC(PCRNAM)
JPILIST(2).RETLENADR  = %LOC(PCRNAM_LEN)
JPILIST(3).END_LIST   = 0
! Call $GETJPI to get data for this process

STATUS = SYS$GETJPIW (
2          %VAL(1),      ! Event flag 1
2          ,             ! No PID
2          ,             ! No process name
2          JPILIST,     ! Item list
2          IOSB,        ! Always use IOSB with $GETJPI!
2          ,             ! No AST
2          )             ! No AST arg
! Check the status in both STATUS and the IOSB, if
! STATUS is OK then copy IOSB(1) to STATUS

IF (STATUS) STATUS = IOSB(1)

! If $GETJPI worked, display the process, if done then
! prepare to exit, otherwise signal an error

IF (STATUS) THEN
    TYPE 1010, PID, PCRNAM(1:PCRNAM_LEN)
1010    FORMAT (' ',Z8.8,' ',A)
ELSE
    CALL LIB$SIGNAL(%VAL(STATUS))
END IF
END

```

Example 4-2 demonstrates (in FORTRAN) how to use the process name to obtain information about a process.

4-12 System Services Features

Using \$GETJPI Alone

Example 4-2: Using \$GETJPI and the Process Name to Obtain Information About a Process

```
! To find information for a particular process by name,
! substitute this code, which includes a process name,
! to call $GETJPI in Example 4--1

! Call $GETJPI to get data for a named process

STATUS = SYS$GETJPIW (
    2          %VAL(1),      ! Event flag 1
    2          ,            ! No PID
    2          'SMITH_1',   ! Process name
    2          JPILIST,     ! Item list
    2          IOSB,       ! Always use IOSB with $GETJPI!
    2          ,           ! No AST
    2          )           ! No AST arg
```

4.7.2 Requesting Information About All Processes on the Local System

You can use \$GETJPI to perform a wildcard search on all processes on the local system. When the **pidadr** argument is specified as **-1**, \$GETJPI returns requested information for each process that the program has privilege to access. The requested information is returned for one process for each call to \$GETJPI.

To perform a wildcard search, call \$GETJPI in a loop, testing the return status.

When performing wildcard searches, \$GETJPI returns an error status for processes that are inaccessible. When a program that uses a **-1** wildcard checks the status value returned by \$GETJPI, it should test for the following status codes:

Status	Explanation
SS\$_NOMOREPROC	All processes have been returned.
SS\$_NOPRIV	The caller lacks sufficient privilege to examine a process.
SS\$_SUSPENDED	The target process is being deleted or is suspended and cannot return the information.

Example 4-3 is a MACRO program that demonstrates how to use the \$GETJPI **-1** wildcard to search for all processes on the local system.

Example 4-3: Using \$GETJPI to Request Information About All Processes on the Local System

```

        .TITLE      WILDJPI - Wildcard $GETJPI example program
        $JPIDEF                                ; Define $GETJPI item codes
        .PSECT     DATA  RD,WRT,NOEXE
IOSB:   .QUAD     0                                ; Completion status
PID:    .LONG     -1                              ; Wildcard PID initialized to -1
ITEMS:  .WORD     32                              ; Size of user name buffer
        .WORD     JPI$ USERNAME                 ; User name item code
        .ADDRESS  UNAME                         ; Address of user name buffer
        .ADDRESS  UNAMESIZ                      ; Address to return user name size
        .LONG     0                              ; End of list
UNAMEDSC:                                ; Length and address form a string
        ; descriptor for LIB$PUT_OUTPUT
UNAMESIZ: .LONG   0                              ; Buffer for size of user name
        .ADDRESS  UNAME                         ; Address of user name buffer
UNAME:   .BLKB    32                              ; User name buffer
        .PSECT     CODE  EXE,NOWRT
        .ENTRY     START, ^M<>
LOOP:   $GETJPIW_S -                              ; Get information and wait
        EFN=#1, -                                ; - use event flag 1
        PIDADR=PID, -                            ; - use wildcard pid
        ITMLST=ITEMS, -                        ; - address of item list
        IOSB=IOSB                               ; - always use IOSB for status check
        BLBC     R0,10$                          ; If failure in R0, check that status
        MOVZWL   IOSB,R0                         ; If success in R0, then move status
        ; from IOSB to R0 for checks
10$:   BLBS     R0,DISPLAY                       ; If success in both R0 and IOSB,
        ; then display this user name
        CMPW    R0,#SS$_NOPRIV                 ; No privilege for this process?
        BEQL   LOOP                             ; If no privilege, try next process
        CMPW    R0,#SS$_SUSPENDED             ; Process suspended?
        BEQL   LOOP                             ; If yes, try next process
        CMPW    R0,#SS$_NOMOREPROC           ; No more processes?
        BEQL   DONE                             ; If yes, finished
        BRB     ERROR                           ; Otherwise, exit with error code in R0
DISPLAY: PUSHAL  UNAMEDSC                      ; Pass address of the user name descriptor
        CALLS  #1,G^LIB$PUT_OUTPUT            ; Display name on SYS$OUTPUT
        BRB     LOOP                             ; Get the next process
DONE:   MOVL    #SS$_NORMAL,R0                 ; Put success status into R0
ERROR:  $EXIT_S R0                             ; Exit with status in R0
        .END      START

```

4.8 Using \$GETJPI with \$PROCESS_SCAN

Using the \$PROCESS_SCAN system service greatly enhances the power of \$GETJPI. With this combination, you can search for selected groups of processes as well as processes on remote nodes. When you use \$GETJPI alone, you specify the **pidadr** or the **prcnam** to locate information about one process. When you use \$GETJPI with \$PROCESS_SCAN, the **pidctx** generated by \$PROCESS_SCAN is used as the **pidadr** argument to \$GETJPI. This process context allows \$GETJPI to use the selection criteria set up in the call to \$PROCESS_SCAN.

4.8.1 Using the \$PROCESS_SCAN Item List and Item-Specific Flags

\$PROCESS_SCAN uses an item list to specify the selection criteria for the \$GETJPI search.

Each entry in the \$PROCESS_SCAN item list contains the following:

- The attribute of the process to be examined
- The value of the attribute or a pointer to the value
- Item-specific flags to control how to interpret the value

Item-specific flags enable you to control selection information. For example, you can use flags to select only those processes that have attribute values that compare to the value in the item list in the following ways:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal
PSCAN\$M_GEQ	Match if value is greater than or equal to
PSCAN\$M_GTR	Match if value is greater than
PSCAN\$M_LEQ	Match if value is less than or equal to
PSCAN\$M_LSS	Match if value is less than
PSCAN\$M_CASE_BLIND	Match without regard to case of letters
PSCAN\$M_PREFIX_MATCH	Match on the leading substring
PSCAN\$M_WILDCARD	Match string is a wildcard pattern

The PSCAN\$M_OR flag is used to connect identical item codes in an item list. For example, in a program that searches for processes owned by several specified users, each user name must be specified in a separate item list entry. The item list entries are connected with the PSCAN\$M_OR flag as in the following FORTRAN example:

```
PSCANLIST(1).BUFLEN = LEN('SMITH')
PSCANLIST(1).CODE   = PSCAN$USERNAME
PSCANLIST(1).BUFADR = %LOC('SMITH')
PSCANLIST(1).ITMFLAGS = PSCAN$M_OR
PSCANLIST(2).BUFLEN = LEN('JONES')
PSCANLIST(2).CODE   = PSCAN$USERNAME
PSCANLIST(2).BUFADR = %LOC('JONES')
PSCANLIST(2).ITMFLAGS = PSCAN$M_OR
PSCANLIST(3).BUFLEN = LEN('JOHNSON')
PSCANLIST(3).CODE   = PSCAN$USERNAME
PSCANLIST(3).BUFADR = %LOC('JOHNSON')
PSCANLIST(3).ITMFLAGS = 0
PSCANLIST(4).END_LIST = 0
```

Use the PSCAN\$M_WILDCARD flag to specify that a character string is to be treated as a wildcard. For example, if you want to search for all process names that begin with the letter *A* and end with the string *ER*, use the string *A*ER* with the PSCAN\$M_WILDCARD flag. If the PSCAN\$M_WILDCARD flag is not specified, the search looks for the 4-character process name *A*ER*.

The PSCAN\$M_PREFIX_MATCH defines a wildcard search to match the initial characters of a string. For example, to find all process names that start with the letters *AB*, use the string *AB* with the PSCAN\$M_PREFIX_MATCH flag. If you do not specify the PSCAN\$M_PREFIX_MATCH flag, the search looks for a process with the 2-character process name *AB*.

The PSCAN\$M_PREFIX_MATCH flag also allows either the PSCAN\$M_EQL or the PSCAN\$M_NEQ flag to be specified. If you specify PSCAN\$M_NEQ, the service matches those names that do *not* begin with the specified character string.

4.8.2 Requesting Information About Processes That Match One Criterion

You can use \$GETJPI with \$PROCESS_SCAN to search for processes that match an item list with one criterion. For example, if you specify a search for processes owned by one user name, \$GETJPI returns only those processes that match the specified user name.

Example 4-4 demonstrates (in FORTRAN) how to perform a \$PROCESS_SCAN search on the local node to select all processes that are owned by user SMITH.

4-16 System Services Features

Using \$GETJPI with \$PROCESS_SCAN

Example 4-4: Using \$GETJPI and \$PROCESS_SCAN to Select Process Information by User Name

```
PROGRAM PROCESS_SCAN

IMPLICIT NONE                                ! Implicit none

INCLUDE '($jpidef) /nolist'                 ! Definitions for $GETJPI
INCLUDE '($pscandef) /nolist'               ! Definitions for $PROCESS_SCAN
INCLUDE '($ssdef) /nolist'                  ! Definitions for SS_NAMES

STRUCTURE /JPIITMLST/                        ! Structure declaration for
UNION                                        ! $GETJPI item lists
MAP
  INTEGER*2 BUFLen,
2      CODE
  INTEGER*4 BUfADR,
2      RETLenADR
END MAP
MAP                                          ! A longword of 0 terminates
  INTEGER*4 END_LIST                        ! an item list
END MAP
END UNION
END STRUCTURE

STRUCTURE /PSCANITMLST/                     ! Structure declaration for
UNION                                        ! $PROCESS_SCAN item lists
MAP
  INTEGER*2 BUFLen,
2      CODE
  INTEGER*4 BUfADR,
2      ITMFLAGS
END MAP
MAP                                          ! A longword of 0 terminates
  INTEGER*4 END_LIST                        ! an item list
END MAP
END UNION
END STRUCTURE

RECORD /PSCANITMLST/                         ! Declare the item list for
2      PSCANLIST(12)                       ! $PROCESS_SCAN

RECORD /JPIITMLST/                           ! Declare the item list for
2      JPILIST(3)                           ! $GETJPI

INTEGER*4 SYS$GETJPIW,                       ! System service entry points
2      SYS$PROCESS_SCAN

INTEGER*4 STATUS,                            ! Status variable
2      CONTEXT,                             ! Context from $PROCESS_SCAN
2      PID                                  ! PID from $GETJPI

INTEGER*2 IOSB(4)                            ! I/O status block for $GETJPI

CHARACTER*16
2      PRCNAM                               ! Process name from $GETJPI
INTEGER*2 PRCNAM_LEN                         ! Process name length

LOGICAL*4 DONE                               ! Done with data loop
```

(continued on next page)

Example 4-4 (Cont.): Using \$GETJPI and \$PROCESS_SCAN to Select Process Information by User Name

```

|*****
!* Initialize item list for $PROCESS_SCAN *
|*****

! Look for processes owned by user SMITH

PSCANLIST(1).BUFLEN  = LEN('SMITH')
PSCANLIST(1).CODE    = PSCAN$_USERNAME
PSCANLIST(1).BUFADR  = %LOC('SMITH')
PSCANLIST(1).ITMFLAGS = 0
PSCANLIST(2).END_LIST = 0
|*****
!*      End of item list initialization      *
|*****

STATUS = SYS$PROCESS_SCAN (          ! Set up the scan context
2                                CONTEXT,
2                                PSCANLIST)

IF (.NOT. STATUS) CALL LIB$SIGNAL (%VAL(STATUS))

! Loop calling $GETJPI with the context

DONE = .FALSE.
DO WHILE (.NOT. DONE)

    ! Initialize $GETJPI item list

        JPILIST(1).BUFLEN  = 4
        JPILIST(1).CODE    = JPI$_PID
        JPILIST(1).BUFADR  = %LOC(PID)
        JPILIST(1).RETLENADR = 0
        JPILIST(2).BUFLEN  = LEN(PCRCNAM)
        JPILIST(2).CODE    = JPI$_PCRCNAM
        JPILIST(2).BUFADR  = %LOC(PCRCNAM)
        JPILIST(2).RETLENADR = %LOC(PCRCNAM_LEN)
        JPILIST(3).END_LIST = 0

    ! Call $GETJPI to get the next SMITH process

        STATUS = SYS$GETJPIW (
2                                %VAL(1), ! Event flag 1
2                                CONTEXT, ! Process context
2                                ,        ! No process name
2                                JPILIST, ! Item list
2                                IOSB,    ! Always use IOSB with $GETJPI!
2                                ,        ! No AST
2                                )        ! No AST arg

        ! Check the status in both STATUS and the IOSB, if
        ! STATUS is OK then copy IOSB(1) to STATUS

        IF (STATUS) STATUS = IOSB(1)

        ! If $GETJPI worked, display the process, if done then
        ! prepare to exit, otherwise signal an error

```

4-18 System Services Features

Using \$GETJPI with \$PROCESS_SCAN

Example 4-4 (Cont.): Using \$GETJPI and \$PROCESS_SCAN to Select Process Information by User Name

```

                IF (STATUS) THEN
                    TYPE 1010, PID, PRCNAM(1:PRCNAM_LEN)
1010             FORMAT (' ',Z8.8,' ',A)
                ELSE IF (STATUS .EQ. SS$_NOMOREPROC) THEN
                    DONE = .TRUE.
                ELSE
                    CALL LIB$SIGNAL(%VAL(STATUS))
                END IF
            END DO
        END
```

4.8.3 Requesting Information About Processes That Match Multiple Values for One Criterion

\$PROCESS_SCAN can also search for processes that match one of a number of values for a single criterion, for example, processes owned by several specified users.

Each value must be specified in a separate item list entry, and the item list entries must be connected with the PSCAN\$M_OR item-specific flag. \$GETJPI selects each process that matches any of the item values.

For example, to look for processes with user names SMITH, JONES, or JOHNSON, substitute FORTRAN code such as that shown in Example 4-5 to initialize the item list in Example 4-4.

Example 4-5: Using \$GETJPI and \$PROCESS_SCAN with Multiple Values for One Criterion

```

!*****
!* Initialize item list for $PROCESS_SCAN *
!*****

! Look for users SMITH, JONES and JOHNSON

PSCANLIST(1).BUFLEN = LEN('SMITH')
PSCANLIST(1).CODE = PSCAN$USERNAME
PSCANLIST(1).BUFADR = %LOC('SMITH')
PSCANLIST(1).ITMFLAGS = PSCAN$M_OR
PSCANLIST(2).BUFLEN = LEN('JONES')
PSCANLIST(2).CODE = PSCAN$USERNAME
PSCANLIST(2).BUFADR = %LOC('JONES')
PSCANLIST(2).ITMFLAGS = PSCAN$M_OR
PSCANLIST(3).BUFLEN = LEN('JOHNSON')
PSCANLIST(3).CODE = PSCAN$USERNAME
PSCANLIST(3).BUFADR = %LOC('JOHNSON')
PSCANLIST(3).ITMFLAGS = 0
PSCANLIST(4).END_LIST = 0

!*****
!* End of item list initialization *
!*****

```

4.8.4 Requesting Information About Processes That Match Multiple Criteria

\$PROCESS_SCAN can be used to search for processes that match values for more than one criterion. When multiple criteria are used, a process must match at least one value for each specified criterion.

Example 4-6 demonstrates (in FORTRAN) how to find any batch process owned by either SMITH or JONES. The program uses syntax similar to the following logical expression to initialize the item list:

```

((username = "SMITH") OR (username = "JONES"))
AND
(MODE = JPI$K_BATCH)

```

4-20 System Services Features

Using \$GETJPI with \$PROCESS_SCAN

Example 4-6: Selecting Processes That Match Multiple Criteria

```
!*****
!* Initialize item list for $PROCESS_SCAN *
!*****

! Look for BATCH jobs owned by users SMITH and JONES

PSCANLIST(1).BUFLEN = LEN('SMITH')
PSCANLIST(1).CODE = PSCAN$_USERNAME
PSCANLIST(1).BUFADR = %LOC('SMITH')
PSCANLIST(1).ITMFLAGS = PSCAN$_OR
PSCANLIST(2).BUFLEN = LEN('JONES')
PSCANLIST(2).CODE = PSCAN$_USERNAME
PSCANLIST(2).BUFADR = %LOC('JONES')
PSCANLIST(2).ITMFLAGS = 0
PSCANLIST(3).BUFLEN = 0
PSCANLIST(3).CODE = PSCAN$_MODE
PSCANLIST(3).BUFADR = JPI$_K_BATCH
PSCANLIST(3).ITMFLAGS = 0
PSCANLIST(4).END_LIST = 0

!*****
!* End of item list initialization *
!*****
```

See Section 4.10 for more information about \$PROCESS_SCAN item codes and flags.

4.8.5 Specifying a Node as Selection Criterion

Several \$PROCESS_SCAN item codes do not refer to attributes of a process, but to the VAXcluster node on which the target process resides. When \$PROCESS_SCAN encounters an item code that refers to a node attribute, it creates an alphabetized list of node names. \$PROCESS_SCAN then directs \$GETJPI to compare the selection criteria against processes on these nodes.

\$PROCESS_SCAN ignores a node specification if it is running on a node that is not part of a VAXcluster system. For example, if you request that \$PROCESS_SCAN select all nodes with the hardware model name "VAX 6360," this search returns information about local processes on a nonclustered system, even if that system is a MicroVAX.

A remote \$GETJPI operation currently requires the system to send a message to the CLUSTER_SERVER process on the remote node. The CLUSTER_SERVER process then collects the information and returns it to the requesting node. This has several implications for clusterwide searches:

- All remote \$GETJPI operations are asynchronous and must be properly synchronized. Many applications that are not correctly synchronized might seem to work on a single node because some \$GETJPI operations are actually synchronous; however, these applications fail if they attempt to examine processes on remote nodes. For more information on how to synchronize

Using \$GETJPI with \$PROCESS_SCAN

\$GETJPI operations, see the section on synchronizing system service completion in the *Introduction to VMS System Services*.

- The CLUSTER_SERVER process is always a current process, because it is executing on behalf of \$GETJPI.
- Attempts by \$GETJPI to examine a node do not succeed during a brief period between the time a node joins the cluster and the time that the CLUSTER_SERVER process is started. Searches that occur during this period skip such a node. Searches that specify only such a booting node fail with a \$GETJPI status of SS\$_UNREACHABLE.
- SS\$_NOMOREPROC is returned after all processes on all specified nodes have been scanned.

4.8.6 Scanning All Nodes on the Cluster for Processes

\$PROCESS_SCAN can scan the entire cluster for processes. For example, to scan the cluster for all processes owned by SMITH, use FORTRAN code like that in Example 4-7 to initialize the item list to find all processes with a nonzero cluster system identifier (CSID) and a user name of SMITH.

Example 4-7: Searching the Cluster for Process Information

```

!*****
!* Initialize item list for $PROCESS_SCAN *
!*****

! Search the cluster for jobs owned by SMITH

PSCANLIST(1).BUFLEN = 0
PSCANLIST(1).CODE = PSCAN$_NODE_CSID
PSCANLIST(1).BUFADR = 0
PSCANLIST(1).ITMFLAGS = PSCAN$_NEQ
PSCANLIST(2).BUFLEN = LEN('SMITH')
PSCANLIST(2).CODE = PSCAN$_USERNAME
PSCANLIST(2).BUFADR = %LOC('SMITH')
PSCANLIST(2).ITMFLAGS = 0
PSCANLIST(3).END_LIST = 0

!*****
!* End of item list initialization *
!*****

```

4.8.7 Scanning Specific Nodes on the Cluster for Processes

You can specify a list of nodes as well. Example 4-8 demonstrates (in FORTRAN) how to design an item list to search for batch processes on the nodes TIGNES, VALTHO, or 2ALPES.

4-22 System Services Features Using \$GETJPI with \$PROCESS_SCAN

Example 4-8: Searching for Process Information on Specific Nodes in the Cluster

```
!*****
!* Initialize item list for $PROCESS_SCAN *
!*****

! Search for BATCH jobs on nodes TIGNES, VALTHO and 2ALPES

PSCANLIST(1).BUFLEN = LEN('TIGNES')
PSCANLIST(1).CODE   = PSCAN$_NODENAME
PSCANLIST(1).BUFADR = %LOC('TIGNES')
PSCANLIST(1).ITMFLAGS = PSCAN$_M_OR
PSCANLIST(2).BUFLEN = LEN('VALTHO')
PSCANLIST(2).CODE   = PSCAN$_NODENAME
PSCANLIST(2).BUFADR = %LOC('VALTHO')
PSCANLIST(2).ITMFLAGS = PSCAN$_M_OR
PSCANLIST(3).BUFLEN = LEN('2ALPES')
PSCANLIST(3).CODE   = PSCAN$_NODENAME
PSCANLIST(3).BUFADR = %LOC('2ALPES')
PSCANLIST(3).ITMFLAGS = 0
PSCANLIST(4).BUFLEN = 0
PSCANLIST(4).CODE   = PSCAN$_MODE
PSCANLIST(4).BUFADR = JPI$_K_BATCH
PSCANLIST(4).ITMFLAGS = 0
PSCANLIST(5).END_LIST = 0

!*****
!*      End of item list initialization      *
!*****
```

4.8.8 Conducting Multiple Simultaneous Searches with \$PROCESS_SCAN

Only one asynchronous remote \$GETJPI request per \$PROCESS_SCAN context is permitted at a time. If you issue a second \$GETJPI request using a context before a previous remote request using the same context has completed, your process stalls in a resource wait until the previous remote \$GETJPI request completes. This stall in the RWAST state prevents your process from executing in user mode or receiving user-mode ASTs.

If you want to run remote searches in parallel, create multiple contexts by calling \$PROCESS_SCAN once for each context. For example, you can design a program that calls \$GETSYI in a loop to find the nodes in the VAXcluster system and creates a separate \$PROCESS_SCAN context for each remote node. Each of these separate contexts can run in parallel. The DCL command SHOW USERS uses this technique to obtain user information more quickly.

Only requests to remote nodes must wait until the previous search using the same context has completed. If the \$PROCESS_SCAN context specifies the local node, any number of \$GETJPI requests using that context can be executed in parallel (within the limits implied by the process quotas for ASTLM and BYTLM).

NOTE: When you use \$GETJPI to reference remote processes, you must properly synchronize all \$GETJPI calls. See the section on asynchronous service completion in the *Introduction to VMS System Services*. Before VMS Version 5.2, if you did not follow these synchronization rules, your programs might have appeared to run correctly. However, if you attempt to run such improperly synchronized programs using \$GETJPI with \$PROCESS_SCAN with a remote process, your program might attempt to use the data before \$GETJPI has returned it.

To perform a synchronous search, in which the program waits until all requested information is available, use \$GETJPIW with an IOSB argument.

4.9 Programming Considerations for GETJPI\$

The following sections describe some important considerations for programming with \$GETJPI.

4.9.1 Using Item Lists Correctly

When \$GETJPI collects data, it makes multiple passes through the item list. If the item list is self-modifying, that is, if the addresses for the output buffers in the item list point back at the item list, \$GETJPI replaces the item list information with the returned data. Therefore, incorrect data might be read or unexpected errors might occur when \$GETJPI reads the item list again.

The number of passes needed by \$GETJPI depends on which item codes are referenced and the state of the target process. A program that appears to work normally might fail when a system has processes that are swapped out of memory or when a process is on a remote node.

The results from \$GETJPI are unpredictable when an item list has buffer pointers that point back at the item list itself. To prevent confusing errors, Digital recommends that you do not use self-modifying item lists.

4.9.2 Improving Performance by Using Buffered \$GETJPI Operations

To request information about a process located on a remote node, \$GETJPI must send a message to the remote node, wait for the response, and then extract the data from the message received. When you perform a search on a remote system, the program must repeat this sequence for each process that \$GETJPI locates.

To reduce the overhead of such a remote search, use \$PROCESS_SCAN with the PSCAN\$_GETJPI_BUFFER_SIZE item code to specify a buffer size for \$GETJPI. When the buffer size is specified by \$PROCESS_SCAN, \$GETJPI packs information for several processes into one buffer and transmits them in a single message. This reduction in the number of messages improves performance.

4-24 System Services Features

Programming Considerations for GETJPI\$

For example, if the \$GETJPI item list requests 100 bytes of information, you might specify a PSCAN\$_GETJPI_BUFFER_SIZE of 1000 bytes so that the service can place information for at least 10 processes in each message. (\$GETJPI does not send fill data in the message buffer; therefore, it is possible that information for more than 10 processes can be packed into the buffer.)

The \$GETJPI buffer must be large enough to hold the data for at least one process. If the buffer is too small, the error code SS\$_IVBUFLN is returned from the \$GETJPI call.

You do not have to allocate space for the \$GETJPI buffer; buffer space is allocated by \$PROCESS_SCAN as part of the search context that it creates. Because \$GETJPI buffering is transparent to the program that calls \$GETJPI, you do not have to modify the loop that calls \$GETJPI.

If you use PSCAN\$_GETJPI_BUFFER_SIZE with \$PROCESS_SCAN, all calls to \$GETJPI using that context must request the same item code information. Because \$GETJPI collects information for more than one process at a time within its buffers, you cannot change the item codes or the lengths of the buffers in the \$GETJPI item list between calls. \$GETJPI returns the error SS\$_BADPARAM if any item code or buffer length changes between \$GETJPI calls. However, you can change the buffer addresses in the \$GETJPI item list from call to call.

The \$GETJPI buffered operation is not used for searching the local node. When a search specifies both multiple nodes and \$GETJPI buffering, the buffering is used on remote nodes but is ignored on the local node. Example 4-9 demonstrates (in FORTRAN) how to use a \$GETJPI buffer to improve performance.

Example 4-9: Using a \$GETJPI Buffer to Improve Performance

```
!*****
!* Initialize item list for $PROCESS_SCAN *
!*****

! Search for jobs owned by users SMITH and JONES
! across the cluster with $GETJPI buffering
```

(continued on next page)

Example 4-9 (Cont.): Using a \$GETJPI Buffer to Improve Performance

```
PSCANLIST(1).BUFLEN = 0
PSCANLIST(1).CODE = PSCAN$_NODE_CSID
PSCANLIST(1).BUFADR = 0
PSCANLIST(1).ITMFLAGS = PSCAN$_NEQ
PSCANLIST(2).BUFLEN = LEN('SMITH')
PSCANLIST(2).CODE = PSCAN$_USERNAME
PSCANLIST(2).BUFADR = %LOC('SMITH')
PSCANLIST(2).ITMFLAGS = PSCAN$_OR
PSCANLIST(3).BUFLEN = LEN('JONES')
PSCANLIST(3).CODE = PSCAN$_USERNAME
PSCANLIST(3).BUFADR = %LOC('JONES')
PSCANLIST(3).ITMFLAGS = 0
PSCANLIST(4).BUFLEN = 0
PSCANLIST(4).CODE = PSCAN$_GETJPI_BUFFER_SIZE
PSCANLIST(4).BUFADR = 1000
PSCANLIST(4).ITMFLAGS = 0
PSCANLIST(5).END_LIST = 0

!*****
!*      End of item list initialization      *
!*****
```

4.9.3 Meeting Remote \$GETJPI Quota Requirements

A remote \$GETJPI request uses system dynamic memory for messages. System dynamic memory uses the process quota BYTLM. To determine the number of bytes required by a \$GETJPI request:

1. Add:

- The size of the \$PROCESS_SCAN item list
- The total size of all reference buffers for \$PROCESS_SCAN (the sum of all buffer length fields in the item list)
- The size of the \$GETJPI item list
- The size of the \$GETJPI buffer
- The size of the calling process RIGHTSLIST
- Approximately 300 bytes for message overhead

2. Double this total.

The total is doubled because the messages consume system dynamic memory on both the sending node and the receiving node.

4-26 System Services Features

Programming Considerations for GETJPI\$

This formula for BYTLM quota applies to both buffered and nonbuffered \$GETJPI requests. For buffered requests, use the value specified in the \$PROCESS_SCAN item, PSCAN\$_GETJPI_BUFFER_SIZE, as the size of the buffer. For nonbuffered requests, use the total length of all data buffers specified in the \$GETJPI item list as the size of the buffer.

If the BYTLM quota is insufficient, \$GETJPI (not \$PROCESS_SCAN) returns the error SS\$_EXBYTLM.

4.9.4 Using \$GETJPI Control Flags

The JPI\$_GETJPI_CONTROL_FLAGS item code, which is specified in the \$GETJPI item list, provides additional control over \$GETJPI. Therefore, \$GETJPI may be unable to retrieve all the data requested in an item list because JPI\$_GETJPI_CONTROL_FLAGS requests that \$GETJPI not perform certain actions that may be necessary to collect the data. For example, a \$GETJPI control flag may instruct the calling program not to retrieve a process that has been swapped out of the balance set.

If \$GETJPI is unable to retrieve any data item because of the restrictions imposed by the control flags, it returns the data length as zero. To verify that \$GETJPI received a data item, examine the data length to be sure that it is not zero. To make this verification possible, be sure to specify the return length for each item in the \$GETJPI item list when any of the JPI\$_GETJPI_CONTROL_FLAGS flags is used.

Unlike other \$GETJPI item codes, the JPI\$_GETJPI_CONTROL_FLAGS item is an input item. The item list entry should specify a longword buffer. The desired control flags should be set in this buffer.

Because the JPI\$_GETJPI_CONTROL_FLAGS item code tells \$GETJPI how to interpret the item list, it must be the first entry in the \$GETJPI item list. The error code SS\$_BADPARAM is returned if it is not the first item in the list.

The following are the \$GETJPI control flags:

JPI\$_NO_TARGET_INSWAP

When JPI\$_NO_TARGET_INSWAP is specified, \$GETJPI does not retrieve a process that has been swapped out of the balance set. JPI\$_NO_TARGET_INSWAP is used to avoid adding the additional load of swapping processes into a system. For example, this flag is used with SHOW SYSTEM to avoid bringing processes into memory to display their accumulated CPU time.

If you specify JPI\$_NO_TARGET_INSWAP and request information from a process that has been swapped out, the following consequences occur:

- Any data stored in the virtual address space of the process is not accessible.
- Any data stored in the process header (PHD) may not be accessible.

- Any data stored in resident data structures, such as the process control block (PCB) or the job information block (JIB), is accessible.

You must examine the return length of an item to verify that the item was retrieved. The information may be located in a different data structure in another release of VMS.

JPI\$M_NO_TARGET_AST

When JPI\$M_NO_TARGET_AST is specified, \$GETJPI does not deliver a kernel-mode AST to the target process. JPI\$M_NO_TARGET_AST is used to avoid executing a target process in order to retrieve information.

If you specify JPI\$M_NO_TARGET_AST and cannot deliver an AST to a target process, the following consequences occur:

- Any data stored in the virtual address space of the process is not accessible.
- Data stored in system data structures, such as the process header (PHD), the process control block (PCB), or the job information block (JIB), is accessible.

You must examine the return length of an item to verify that the item was retrieved. The information may be located in a different data structure in another release of VMS.

The use of the flag JPI\$M_NO_TARGET_AST also implies that \$GETJPI does not swap in a process, because \$GETJPI would bring a process into memory only to deliver an AST to that process.

JPI\$M_IGNORE_TARGET_STATUS

When JPI\$M_IGNORE_TARGET_STATUS is specified, \$GETJPI attempts to retrieve as much information as possible, even if the process is suspended or being deleted. JPI\$M_IGNORE_TARGET_STATUS is used to retrieve all possible information from a process. For example, this flag is used with SHOW SYSTEM to display processes that are suspended, are being deleted, or are in miscellaneous wait states.

Example 4-10 demonstrates (in FORTRAN) how to use \$GETJPI control flags to avoid swapping processes during a \$GETJPI call.

4-28 System Services Features Programming Considerations for GETJPI\$

Example 4-10: Using \$GETJPI Control Flags to Avoid Swapping a Process into the Balance Set

```
PROGRAM CONTROL_FLAGS
IMPLICIT NONE                                ! Implicit none
INCLUDE '($jpidef) /nolist'                 ! Definitions for $GETJPI
INCLUDE '($pscandef) /nolist'              ! Definitions for $PROCESS_SCAN
INCLUDE '($ssdef) /nolist'                 ! Definitions for SS$_names

STRUCTURE /JPIITMLST/                        ! Structure declaration for
UNION                                        ! $GETJPI item lists
  MAP
    INTEGER*2 BUFLEN,
  2      CODE
    INTEGER*4 BUFADR,
  2      RETLENADR
  END MAP
  MAP                                        ! A longword of 0 terminates
    INTEGER*4 END_LIST                      ! an item list
  END MAP
END UNION
END STRUCTURE

STRUCTURE /PSCANITMLST/                     ! Structure declaration for
UNION                                        ! $PROCESS_SCAN item lists
  MAP
    INTEGER*2 BUFLEN,
  2      CODE
    INTEGER*4 BUFADR,
  2      ITMFLAGS
  END MAP
  MAP                                        ! A longword of 0 terminates
    INTEGER*4 END_LIST                      ! an item list
  END MAP
END UNION
END STRUCTURE
RECORD /PSCANITMLST/                        ! Declare the item list for
  2      PSCANLIST(5)                      ! $PROCESS_SCAN

RECORD /JPIITMLST/                          ! Declare the item list for
  2      JPILIST(6)                        ! $GETJPI

INTEGER*4 SYS$GETJPIW,                      ! System service entry points
  2      SYS$PROCESS_SCAN

INTEGER*4 STATUS,                          ! Status variable
  2      CONTEXT,                         ! Context from $PROCESS_SCAN
  2      PID,                             ! PID from $GETJPI
  2      JPIFLAGS                          ! Flags for $GETJPI

INTEGER*2 IO SB(4)                          ! I/O status block for $GETJPI
```

(continued on next page)

Example 4-10 (Cont.): Using \$GETJPI Control Flags to Avoid Swapping a Process into the Balance Set

```

CHARACTER*16
2      PRCNAM,           ! Process name from $GETJPI
2      NODENAME         ! Node name from $GETJPI
INTEGER*2 PRCNAM_LEN,  ! Process name length
2      NODENAME_LEN    ! Node name length

CHARACTER*80
2      IMAGNAME         ! Image name from $GETJPI
INTEGER*2 IMAGNAME_LEN ! Image name length

LOGICAL*4 DONE         ! Done with data loop

!*****
!*  Initialize item list for $PROCESS_SCAN *
!*****

! Look for interactive and batch jobs across
! the cluster with $GETJPI buffering

PSCANLIST(1).BUFLEN  = 0
PSCANLIST(1).CODE   = PSCAN$_NODE_CSID
PSCANLIST(1).BUFADR = 0
PSCANLIST(1).ITMFLAGS = PSCAN$_M_NEQ
PSCANLIST(2).BUFLEN  = 0
PSCANLIST(2).CODE   = PSCAN$_MODE
PSCANLIST(2).BUFADR = JPI$_K_INTERACTIVE
PSCANLIST(2).ITMFLAGS = PSCAN$_M_OR
PSCANLIST(3).BUFLEN  = 0
PSCANLIST(3).CODE   = PSCAN$_MODE
PSCANLIST(3).BUFADR = JPI$_K_BATCH
PSCANLIST(3).ITMFLAGS = 0
PSCANLIST(4).BUFLEN  = 0
PSCANLIST(4).CODE   = PSCAN$_GETJPI_BUFFER_SIZE
PSCANLIST(4).BUFADR = 1000
PSCANLIST(4).ITMFLAGS = 0
PSCANLIST(5).END_LIST = 0
!*****
!*      End of item list initialization      *
!*****

STATUS = SYS$PROCESS_SCAN (      ! Set up the scan context
2      CONTEXT,
2      PSCANLIST)

IF (.NOT. STATUS) CALL LIB$SIGNAL (%VAL(STATUS))

! Initialize $GETJPI item list
  
```

(continued on next page)

4-30 System Services Features

Programming Considerations for GETJPI\$

Example 4-10 (Cont.): Using \$GETJPI Control Flags to Avoid Swapping a Process into the Balance Set

```
JPILIST(1).BUFLEN      = 4
JPILIST(1).CODE       = IAND ('FFFF'X, JPI$GETJPI_CONTROL_FLAGS)
JPILIST(1).BUFADR     = %LOC(JPIFLAGS)
JPILIST(1).RETLENADR  = 0
JPILIST(2).BUFLEN     = 4
JPILIST(2).CODE       = JPI$_PID
JPILIST(2).BUFADR     = %LOC(PID)
JPILIST(2).RETLENADR  = 0
JPILIST(3).BUFLEN     = LEN(PCRNAM)
JPILIST(3).CODE       = JPI$_PCRNAM
JPILIST(3).BUFADR     = %LOC(PCRNAM)
JPILIST(3).RETLENADR  = %LOC(PCRNAM_LEN)
JPILIST(4).BUFLEN     = LEN(IMAGNAME)
JPILIST(4).CODE       = JPI$_IMAGNAME
JPILIST(4).BUFADR     = %LOC(IMAGNAME)
JPILIST(4).RETLENADR  = %LOC(IMAGNAME_LEN)
JPILIST(5).BUFLEN     = LEN(NODENAME)
JPILIST(5).CODE       = JPI$_NODENAME
JPILIST(5).BUFADR     = %LOC(NODENAME)
JPILIST(5).RETLENADR  = %LOC(NODENAME_LEN)
JPILIST(6).END_LIST   = 0
! Loop calling $GETJPI with the context

DONE = .FALSE.
JPIFLAGS = IOR (JPI$_NO_TARGET_INSWAP, JPI$_IGNORE_TARGET_STATUS)
DO WHILE (.NOT. DONE)

    ! Call $GETJPI to get the next process

    STATUS = SYS$GETJPIW (
2          %VAL(1),      ! Event flag 1
2          CONTEXT,     ! Process context
2          ,             ! No process name
2          JPILIST,     ! Itemlist
2          IOSB,        ! Always use IOSB with $GETJPI!
2          ,             ! No AST
2          )            ! No AST arg

    ! Check the status in both STATUS and the IOSB, if
    ! STATUS is OK then copy IOSB(1) to STATUS

    IF (STATUS) STATUS = IOSB(1)

    ! If $GETJPI worked, display the process, if done then
    ! prepare to exit, otherwise signal an error
```

(continued on next page)

Example 4-10 (Cont.): Using \$GETJPI Control Flags to Avoid Swapping a Process into the Balance Set

```

                IF (STATUS) THEN
                    IF (IMAGNAME_LEN .EQ. 0) THEN
                        TYPE 1010, PID, NODENAME, PRCNAM
                    ELSE
2                 TYPE 1020, PID, NODENAME, PRCNAM,
                    IMAGNAME(1:IMAGNAME_LEN)
                END IF
                ELSE IF (STATUS .EQ. SS$_NOMOREPROC) THEN
                    DONE = .TRUE.
                ELSE
                    CALL LIB$SIGNAL(%VAL(STATUS))
                END IF
            END DO
1010    FORMAT (' ',Z8.8,' ',A6,':: ',A,' (no image)')
1020    FORMAT (' ',Z8.8,' ',A6,':: ',A,' ',A)
END
```

4.10 \$PROCESS_SCAN and \$DEVICE_SCAN System Services

This section contains reference information for two system services, \$DEVICE_SCAN, an input/output system service, and \$PROCESS_SCAN, a process information system service.

\$DEVICE_SCAN—Scan for Devices

The Device Scan system service returns the names of all devices that match a specified set of search criteria. The names returned by \$DEVICE_SCAN can then be passed to another service, for example, \$GETDVI or \$MOUNT.

format

SYS\$DEVICE_SCAN *return_devnam ,retlen ,[search_devnam] ,[itmlst] ,[contxt]*

returns:

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Longword condition value. All system services (except \$EXIT) return by immediate value a condition value in R0. Condition values that this service returns are listed under “condition values returned.”

arguments

return_devnam

VMS Usage: **char_string**
type: **character-coded text string**
access: **write only**
mechanism: **by descriptor—fixed length string descriptor**

Buffer to receive the device name. The **return_devnam** argument is the address of a character string descriptor pointing to a buffer into which \$DEVICE_SCAN writes the name of the first or next device that matches the specified search criteria. The maximum size of any device name is 64 bytes.

retlen

VMS Usage: **word_unsigned**
type: **word (unsigned)**
access: **write only**
mechanism: **by reference**

Length of the device name string returned by \$DEVICE_SCAN. The **retlen** argument is the address of a word into which \$DEVICE_SCAN writes the length of the device name string.

search_devnam

VMS Usage: **device_name**
type: **character-coded text string**
access: **read only**
mechanism: **by descriptor—fixed length string descriptor**

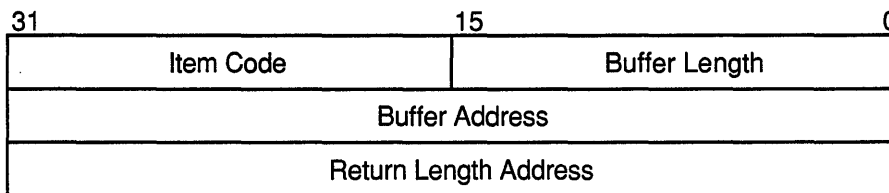
Name of the device for which \$DEVICE_SCAN is to search. The **search_devnam** argument accepts the standard wildcard characters, the asterisk (*), which matches any sequence of characters, and the percent sign (%), which matches any one character. For example, to match all unit 0 DU devices on any controller, specify *DU%0. This string is compared to the most complete device name (DVI\$_ALLDEVNAM).

itmlst

VMS Usage: **item_list_3**
 type: **longword_unsigned**
 access: **read only**
 mechanism: **by reference**

Item list specifying search criteria used to identify the device names for return by \$DEVICE_SCAN. The **itmlst** argument is the address of a list of item descriptors, each of which describes one search criterion. The list of item descriptors is terminated by a longword of 0.

The following figure depicts the format of a single item descriptor:



ZK-1705-GE

\$DEVICE_SCAN Item Descriptor Fields

buffer length

A word containing a user-supplied integer specifying the length (in bytes) of the buffer from which \$DEVICE_SCAN is to read the information. The length of the buffer needed depends upon the item code specified in the item code field of the item descriptor.

item code

A word containing a user-supplied symbolic code specifying the item of information that \$DEVICE_SCAN is to return. The \$DVSDEF macro defines these codes. Each item code is described following this list of item descriptor fields.

buffer address

A longword containing the user-supplied address of the buffer from which \$DEVICE_SCAN is to read the information.

4-34 System Services Features

\$DEVICE_SCAN

return length address

This field is not currently used.

\$DEVICE_SCAN Item Codes

DVS\$_DEVCLASS

An input value item code that specifies, as an unsigned longword, the device class being searched. The \$DCDEF macro defines these classes.

The DVS\$_DEVCLASS argument is a longword containing this number; however, DVS\$_DEVCLASS uses only the low-order byte of the longword.

DVS\$_DEVTYPE

An input value item code that specifies, as an unsigned longword, the device type for which \$DEVICE_SCAN is going to search. The \$DCDEF macro defines these types.

The DVS\$_DEVTYPE argument is a longword containing this number; however, DVS\$_DEVTYPE uses only the low-order byte of the longword. DVS\$_DEVTYPE should be used with \$DVS_DEVCLASS to specify the device type being searched for.

contxt

VMS Usage: **quadword_unsigned**
type: **quadword (unsigned)**
access: **modify**
mechanism: **by reference**

Value used to indicate the current position of a \$DEVICE_SCAN search. The **contxt** argument is the address of the quadword that receives this information. On the initial call, the quadword should contain 0.

description

The Device Scan service returns all device names that match a specified set of search criteria. The device names are returned for one process per call. A context value is used to continue multiple calls to \$DEVICE_SCAN.

\$DEVICE_SCAN allows wildcard searches based on device names, device classes, and device types. It also provides the ability to perform a wildcard search on other device-related services.

\$DEVICE_SCAN makes it possible to combine search criteria. For example, to find only RA82 devices use the following selection criteria:

```
DVS$_DEVCLASS = DC$_DISK and DVS$_DEVTYPE = DT$_RA82
```

To find all mailboxes with *MB* as part of the device name (excluding mailboxes such as NLA0), use the following selection criteria:

```
DVS$_DEVCLASS = DC$_MAILBOX and DEVNAM = *MB*
```

condition values returned

SS\$NORMAL

The service completed successfully.

SS\$ACCVIO

The **search_devnam**, **itmlst**, or **contxt** argument cannot be read by the caller, or the **retlen**, **return_devnam**, or **contxt** argument cannot be written by the caller.

SS\$BADPARAM

The **contxt** argument contains an invalid value or the item list contains an invalid item code.

SS\$NOSUCHDEV

The specified device does not exist on the host system.

SS\$NOMOREDEV

No more devices match the specified search criteria.

\$PROCESS_SCAN—Process Scan

The Process Scan system service creates and initializes a process context that is used by \$GETJPI to scan processes on the local system or across the nodes in a VAXcluster system. An item list is used to specify selection criteria to obtain information about specific processes, for example, all processes owned by one user or all batch processes.

format

SYS\$PROCESS_SCAN *pidctx* [,*itmlst*]

returns:

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Longword condition value. All system services (except \$EXIT) return by immediate value a condition value in R0. Condition values that this service returns are listed under “condition values returned.”

arguments

pidctx

VMS Usage: **process_id**
type: **longword (unsigned)**
access: **modify**
mechanism: **by reference**

Context value supplied by \$PROCESS_SCAN to be used as the **pidadr** argument of \$GETJPI. The **pidctx** argument is the address of a longword that is to receive the process context longword. This longword normally contains zero or a previous context. If it contains a previous context, the old context is deleted. If it contains a value other than zero or a previous context, the old value is ignored.

itmlst

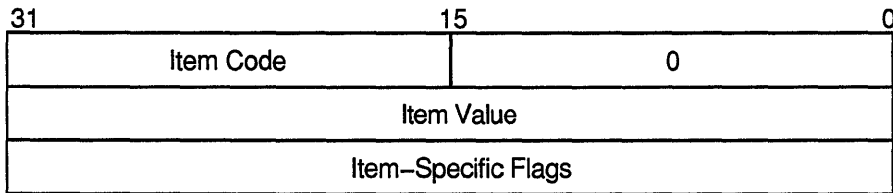
VMS Usage: **item_list_3**
type: **longword (unsigned)**
access: **read only**
mechanism: **by reference**

Item list specifying selection criteria to be used by the scan or to control the scan.

The **itmlst** argument is the address of a list of item descriptors, each of which describes one selection criterion or control option. Within each selection criterion you can include several item entries. The list of item descriptors is terminated by a longword of zero.

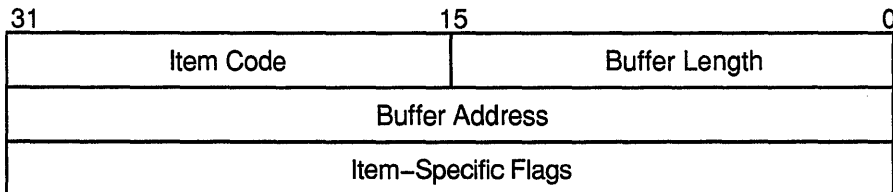
The information in the item list is passed to the item descriptor in one of two ways. If the item descriptor can always hold the actual value of the selection criterion, the value is placed in the second longword of the item descriptor and the buffer length is specified as zero. If the item descriptor points to the actual value of the selection criterion, the address of the value is placed in the second longword of the item descriptor and you must specify the buffer length for the selection criterion. Each item code description specifies whether the information is passed by value or by reference.

The following figure depicts the format of an item descriptor that passes the selection criterion as a value:



ZK-0949A-GE

The following figure depicts the format of an item descriptor that passes the selection criterion by reference:



ZK-0948A-GE

\$PROCESS_SCAN Item Descriptor Fields

buffer address

A longword containing the user-supplied address of the buffer from which \$PROCESS_SCAN retrieves information needed by the scan. When you specify an item code that is passed by reference, \$PROCESS_SCAN uses the address as a pointer to the actual value. See the description of the **item value** field for information about item codes that are passed by value.

buffer length

Buffer length is specified in a different way for the two types of item descriptors:

- Character string or reference descriptors
A word containing a user-supplied integer specifying the length (in bytes) of the buffer from which \$PROCESS_SCAN retrieves a selection criterion. The length of the buffer needed depends upon the item code specified in the item descriptor.

- Immediate value descriptors

The length of the buffer is always specified as zero.

item code

A word containing the selection criterion. These codes are defined by the \$PSCANDEF macro. Each item code is described after this list of descriptor fields.

item value

A longword containing the actual value of the selection criterion. When you specify an item code that is passed by value, \$PROCESS_SCAN searches for the actual value contained in the item list. See the description of the **buffer address** field for information about item codes that are passed by reference.

item-specific flags

A longword that contains flags to help control selection information. Item-specific flags, for example EQL or NEQ, are used to specify how the value specified in the item descriptor is compared to the process value.

These flags are defined by the \$PSCANDEF macro. Some flags are common to multiple item codes; other flags are specific to an individual item code. See the description of each item code to determine which flags are used.

For item codes that describe bit masks or character strings, these flags control how the bit mask or character string is compared with that in the process. By default, they are compared for equality.

For item codes that describe integers, these flags specify an arithmetic comparison of an integer item with the process attribute. For example, a PSCAN\$M_GTR selection specifying the value 4 for the item code PSCAN\$_PRIB finds only the processes with a base priority above 4. Without one of these flags, the comparison is for equality.

\$PROCESS_SCAN Item Codes

PSCAN\$_ACCOUNT

When you specify `PSCAN$_ACCOUNT`, `$GETJPI` returns information about processes that match the account field.

If the string supplied in the item descriptor is shorter than the account field, the string is padded with blanks for the comparison unless the item-specific flag `PSCAN$_M_PREFIX_MATCH` is present.

Because the information is a character string, the selection value is passed by reference. The length of the buffer is placed in the first word of the item descriptor and the address of the buffer is placed in the second longword.

Although the current length of the account field is eight bytes, the `PSCAN$_ACCOUNT` buffer can be up to 64 bytes in length. If the buffer length is zero or greater than 64, the `SS$_IVBUFLLEN` error is returned.

The following flags can be used with this item code:

Item-Specific Flag	Description
<code>PSCAN\$_M_OR</code>	Match this value or the next value
<code>PSCAN\$_M_EQL</code>	Match value exactly (the default)
<code>PSCAN\$_M_NEQ</code>	Match if value is not equal
<code>PSCAN\$_M_CASE_BLIND</code>	Match without regard to case of letters
<code>PSCAN\$_M_PREFIX_MATCH</code>	Match on the leading substring
<code>PSCAN\$_M_WILDCARD</code>	Match string is a wildcard pattern

PSCAN\$_AUTHPRI

When you specify `PSCAN$_AUTHPRI`, `$GETJPI` returns information about processes that match the authorized base priority field.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
<code>PSCAN\$_M_OR</code>	Match this value or the next value
<code>PSCAN\$_M_EQL</code>	Match value exactly (the default)
<code>PSCAN\$_M_NEQ</code>	Match if value is not equal

4-40 System Services Features

\$PROCESS_SCAN

Item-Specific Flag	Description
PSCAN\$M_GEQ	Match if value is greater than or equal to
PSCAN\$M_GTR	Match if value is greater than
PSCAN\$M_LEQ	Match if value is less than or equal to
PSCAN\$M_LSS	Match if value is less than

PSCAN\$_CURPRIV

When you specify PSCAN\$_CURPRIV, \$GETJPI returns information about processes that match the current privilege field. Privilege bits are defined by the \$PRVDEF macro.

Because the bit mask information is too long to be passed by value, the information is passed by reference. The privilege buffer must be exactly eight bytes; otherwise, the SS\$_IVBUFLN error is returned.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQ	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal
PSCAN\$M_BIT_ALL	All bits set in pattern set in target
PSCAN\$M_BIT_ANY	Any bit set in pattern set in target

PSCAN\$_GETJPI_BUFFER_SIZE

When you specify PSCAN\$_GETJPI_BUFFER_SIZE, you determine the size of a buffer to be used by \$GETJPI to process multiple requests in a single message. Using this item code can greatly improve the performance of scans on remote nodes because fewer messages are needed. This item code is ignored during scans on the local node.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero. The buffer is allocated by \$PROCESS_SCAN; you do not have to allocate a buffer.

If you use PSCAN\$_GETJPI_BUFFER_SIZE with \$PROCESS_SCAN, all calls to \$GETJPI using the context established by \$PROCESS_SCAN must request the same item code information. Because \$GETJPI locates information for more than one process at a time, it is not possible to change the item codes or the length of the buffers used in the \$GETJPI item list. \$GETJPI checks each call and returns the error SS\$_BADPARAM if an attempt is made to change the item list during

a buffered process scan. However, the buffer addresses can be changed between \$GETJPI calls.

Because the locating and buffering of information by \$GETJPI is transparent to a calling program, you are not required to change the way \$GETJPI is called when you use this item code.

The \$GETJPI buffer uses the process quota BYTLM. If the buffer is too large for the process quota, \$GETJPI (not \$PROCESS_SCAN) returns the error SS\$_EXBYTLM. If the buffer specified is not large enough to contain the data for at least one process, \$GETJPI returns the error SS\$_BADPARAM.

No item-specific flags are used with PSCAN\$_GETJPI_BUFFER_SIZE.

PSCAN\$_GRP

When you specify PSCAN\$_GRP, \$GETJPI returns information about processes that match the UIC group number.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. Because the value of the group number is a word, the high-order word of the value is ignored. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$_M_OR	Match this value or the next value
PSCAN\$_M_EQL	Match value exactly (the default)
PSCAN\$_M_NEQ	Match if value is not equal
PSCAN\$_M_GEQ	Match if value is greater than or equal to
PSCAN\$_M_GTR	Match if value is greater than
PSCAN\$_M_LEQ	Match if value is less than or equal to
PSCAN\$_M_LSS	Match if value is less than

PSCAN\$_HW_MODEL

When you specify PSCAN\$_HW_MODEL, \$GETJPI returns information about processes that match the specified CPU hardware model number.

The hardware model number is an integer, such as VAX\$K_V8840. The VAX\$ symbols are defined by the \$VAXDEF macro.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

4-42 System Services Features

\$PROCESS_SCAN

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal

PSCAN\$_HW_NAME

When you specify PSCAN\$_HW_NAME, \$GETJPI returns information about processes that match the specified CPU hardware name, such as VAX 11/780, VAX 8800, or VAXstation II/GPX.

Because the information is a character string, the selection value is passed by reference. The length of the selection value is placed in the first word of the item descriptor and the address of the buffer is placed in the second longword.

The PSCAN\$_HW_NAME buffer can be up to 128 bytes in length. If the buffer length is zero or greater than 128, the SS\$_IVBUFLen error is returned.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal
PSCAN\$M_CASE_BLIND	Match without regard to case of letters
PSCAN\$M_PREFIX_MATCH	Match on the leading substring
PSCAN\$M_WILDCARD	Match a wildcard pattern

PSCAN\$_JOBPRCNT

When you specify PSCAN\$_JOBPRCNT, \$GETJPI returns information about processes that match the subprocess count for the job (the count of all subprocesses in the job tree).

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal
PSCAN\$M_GEQ	Match if value is greater than or equal to
PSCAN\$M_GTR	Match if value is greater than
PSCAN\$M_LEQ	Match if value is less than or equal to
PSCAN\$M_LSS	Match if value is less than

PSCAN\$_JOBTYPE

When you specify PSCAN\$_JOBTYPE, \$GETJPI returns information about processes that match the job type. The job type values include the following:

Value	Description
JPI\$K_LOCAL	Local interactive process
JPI\$K_DIALUP	Interactive process accessed by a modem line
JPI\$K_REMOTE	Interactive process accessed by using SET HOST
JPI\$K_BATCH	Batch process
JPI\$K_NETWORK	Noninteractive network process
JPI\$K_DETACHED	Detached process

These values are defined by the \$JPIDEF macro. Note that jobtype values are similar to mode values. See PSCAN\$_MODE.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal

4-44 System Services Features

\$PROCESS_SCAN

PSCAN\$_MASTER_PID

When you specify `PSCAN$_MASTER_PID`, `$GETJPI` returns information about processes that are descendants of the specified parent process. The master process is the first process created in the job tree. The `PSCAN$_OWNER` item is similar, but the owner process is the process that created the target process (the owner process might itself be a subprocess). Although all jobs in a job tree must have the same master, they may have different owners.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
<code>PSCAN\$_M_OR</code>	Match this value or the next value
<code>PSCAN\$_M_EQL</code>	Match value exactly (the default)
<code>PSCAN\$_M_NEQ</code>	Match if value is not equal

PSCAN\$_MEM

When you specify `PSCAN$_MEM`, `$GETJPI` returns information about processes that match the UIC member number.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. Because the value of the member number is a word, the high-order word of the value is ignored. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
<code>PSCAN\$_M_OR</code>	Match this value or the next value
<code>PSCAN\$_M_EQL</code>	Match value exactly (the default)
<code>PSCAN\$_M_NEQ</code>	Match if value is not equal
<code>PSCAN\$_M_GEQ</code>	Match if value is greater than or equal to
<code>PSCAN\$_M_GTR</code>	Match if value is greater than
<code>PSCAN\$_M_LEQ</code>	Match if value is less than or equal to
<code>PSCAN\$_M_LSS</code>	Match if value is less than

PSCAN\$_MODE

When you specify `PSCAN$_MODE`, `$GETJPI` returns information about processes that match the specified mode. Mode values include the following:

Value	Description
JPI\$K_INTERACTIVE	Interactive process
JPI\$K_BATCH	Batch job
JPI\$K_NETWORK	Noninteractive network job
JPI\$K_OTHER	Detached and other process

These values are defined by the \$JPIDEF macro. Note that values checked by PSCAN\$_MODE are similar to PSCAN\$_JOBTYPE values.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$_M_OR	Match this value or the next value
PSCAN\$_M_EQL	Match value exactly (the default)
PSCAN\$_M_NEQ	Match if value is not equal

PSCAN\$_NODE_CSID

When you specify PSCAN\$_NODE_CSID, \$GETJPI returns information about processes on the specified nodes. To scan all nodes in a VAXcluster system, you specify a CSID of zero and the item-specific flag PSCAN\$_M_NEQ.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$_M_OR	Match this value or the next value
PSCAN\$_M_EQL	Match value exactly (the default)
PSCAN\$_M_NEQ	Match if value is not equal

PSCAN\$_NODENAME

When you specify PSCAN\$_NODENAME, \$GETJPI returns information about processes that match the specified node names.

4-46 System Services Features

\$PROCESS_SCAN

To scan all of the nodes in a VAXcluster system, specify the node name using an asterisk wildcard (*) and the PSCAN\$M_WILDCARD item-specific flag.

Because the information is a character string, the selection value is passed by reference. The length of the selection value is placed in the first word of the item descriptor and the address of the buffer is placed in the second longword.

Although the current length of the node name is 6 bytes, the PSCAN\$_NODENAME buffer can be up to 64 bytes in length. If the buffer length is zero or greater than 64, the SS\$_IVBUFLLEN error is returned.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal
PSCAN\$M_CASE_BLIND	Match without regard to case of letters
PSCAN\$M_PREFIX_MATCH	Match on leading substring
PSCAN\$M_WILDCARD	Match a wildcard pattern

PSCAN\$_OWNER

When you specify PSCAN\$_OWNER, \$GETJPI returns information about processes that are immediate descendants of the specified process. The PSCAN\$_MASTER_PID item is similar, but the owner process is the process that created the target process (the owner process might itself be a subprocess). Although all jobs in a job tree must have the same master, they can have different owners.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal

\$PROCESS_SCAN***PSCAN\$_PRCNT***

When you specify `PSCAN$_PRCNT`, `$GETJPI` returns information about processes that match the subprocess count (the count of all immediate descendants of a given process). The `PSCAN$_JOBPRCNT` item code is similar, except that `JOBPRCNT` is the count of all subprocesses in a job.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
<code>PSCAN\$_M_OR</code>	Match this value or the next value
<code>PSCAN\$_M_EQL</code>	Match value exactly (the default)
<code>PSCAN\$_M_NEQ</code>	Match if value is not equal
<code>PSCAN\$_M_GEQ</code>	Match if value is greater than or equal to
<code>PSCAN\$_M_GTR</code>	Match if value is greater than
<code>PSCAN\$_M_LEQ</code>	Match if value is less than or equal to
<code>PSCAN\$_M_LSS</code>	Match if value is less than

PSCAN\$_PRCNAM

When you specify `PSCAN$_PRCNAM`, `$GETJPI` returns information about processes that match the specified process names.

The process name string is padded with blanks for the comparison unless the item-specific flag `PSCAN$_M_PREFIX_MATCH` is present.

Because the information is a character string, the selection value is passed by reference. The length of the selection value is placed in the first word of the item descriptor and the address of the buffer is placed in the second longword.

Although the current length of the process name field is 15 bytes, the `PSCAN$_PRCNAM` buffer can be up to 64 bytes in length. If the buffer length is zero or greater than 64, the `SS$_IVBUFLN` error is returned.

The following flags can be used with this item code:

Item-Specific Flag	Description
<code>PSCAN\$_M_OR</code>	Match this value or the next value
<code>PSCAN\$_M_EQL</code>	Match value exactly (the default)

4-48 System Services Features

\$PROCESS_SCAN

Item-Specific Flag	Description
PSCAN\$M_NEQ	Match if value is not equal
PSCAN\$M_CASE_BLIND	Match without regard to case of letters
PSCAN\$M_PREFIX_MATCH	Match on leading substring
PSCAN\$M_WILDCARD	Match a wildcard pattern

PSCAN\$ PRI

When you specify PSCAN\$ PRI, \$GETJPI returns information about processes that match current priority. Note that the current priority of a process may be temporarily increased as a result of system events such as the completion of I/O.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal
PSCAN\$M_GEQ	Match if value is greater than or equal to
PSCAN\$M_GTR	Match if value is greater than
PSCAN\$M_LEQ	Match if value is less than or equal to
PSCAN\$M_LSS	Match if value is less than

PSCAN\$ PRIB

When you specify PSCAN\$ PRIB, \$GETJPI returns information about processes that match base priority.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)

Item-Specific Flag	Description
PSCAN\$M_NEQ	Match if value is not equal
PSCAN\$M_GEQ	Match if value is greater than or equal to
PSCAN\$M_GTR	Match if value is greater than
PSCAN\$M_LEQ	Match if value is less than or equal to
PSCAN\$M_LSS	Match if value is less than

PSCAN\$_STATE

When you specify PSCAN\$_STATE, \$GETJPI returns information about processes that match the specified process state. State values, for example SCH\$C_COM and SCH\$C_PFW, are defined by the \$STATEDEF macro.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal

PSCAN\$_STS

When you specify PSCAN\$_STS, \$GETJPI returns information that matches the current status mask. Without any item-specific flags, the match is for a process mask that is equal to the pattern. Status bits, for example PCB\$V_ASTPEN or PCB\$V_PSWAPM, are defined by the \$PCBDEF macro.

This bit mask item code uses an immediate value descriptor; the selection value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal

4-50 System Services Features

\$PROCESS_SCAN

Item-Specific Flag	Description
PSCAN\$M_BIT_ALL	All bits set in pattern set in target
PSCAN\$M_BIT_ANY	Any bit set in pattern set in target

PSCAN\$_TERMINAL

When you specify PSCAN\$_TERMINAL, \$GETJPI returns information that matches the specified terminal names. The terminal name string is padded with blanks for the comparison unless the item-specific flag PSCAN\$M_PREFIX_MATCH is present.

Because the information is a character string, the selection value is passed by reference. The length of the selection value is placed in the first word of the item descriptor and the address of the buffer is placed in the second longword.

Although the current length of the terminal name field is 8 bytes, the PSCAN\$_TERMINAL buffer can be up to 64 bytes in length. If the buffer length is zero or greater than 64, the SS\$_IVBUFLen error is returned.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal
PSCAN\$M_CASE_BLIND	Match without regard to case of letters
PSCAN\$M_PREFIX_MATCH	Match on leading substring
PSCAN\$M_WILDCARD	Match a wildcard pattern

PSCAN\$_UIC

When you specify PSCAN\$_UIC, \$GETJPI returns information about processes that match the UIC identifier. To convert an alphanumeric identifier name to the internal identifier, use the \$ASCTOID system service before calling \$PROCESS_SCAN.

This integer item code is passed by value; the value is placed in the second longword of the item descriptor. The buffer length must be specified as zero.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal

PSCAN\$_USERNAME

When you specify PSCAN\$_USERNAME, \$GETJPI returns information about processes that match the specified user name.

The user name string is padded with blanks for the comparison unless the item-specific flag PSCAN\$M_PREFIX_MATCH is present.

Because the information is a character string, the selection value is passed by reference. The length of the selection value is placed in the first word of the item descriptor and the address of the buffer is placed in the second longword.

Although the current length of the username field is 12 bytes, the PSCAN\$_USERNAME buffer can be up to 64 bytes in length. If the buffer length is zero or greater than 64, the SS\$_IVBUFLN error is returned.

The following flags can be used with this item code:

Item-Specific Flag	Description
PSCAN\$M_OR	Match this value or the next value
PSCAN\$M_EQL	Match value exactly (the default)
PSCAN\$M_NEQ	Match if value is not equal
PSCAN\$M_CASE_BLIND	Match without regard to case of letters
PSCAN\$M_PREFIX_MATCH	Match on leading substring
PSCAN\$M_WILDCARD	Match a wildcard pattern

The following flags can be furnished in the item-specific flag field of the item descriptor.

\$PROCESS_SCAN Item-Specific Flags

PSCAN\$M_BIT_ALL

If the PSCAN\$M_BIT_ALL flag is used, all bits set in the pattern mask specified by the item descriptor must also be set in the process mask. Other bits in the process mask may also be set.

For item codes that describe bit masks, such as privilege masks and status words, this flag controls how the pattern bit mask specified by the item descriptor is compared with that in the process. By default, the bit masks are compared for equality.

4-52 System Services Features

\$PROCESS_SCAN

The `PSCAN$M_BIT_ALL` flag is used only with bit masks.

PSCAN\$M_BIT_ANY

If the `PSCAN$M_BIT_ANY` flag is used, a match occurs if any bit in the pattern mask is also set in the process mask.

For item codes that describe bit masks, such as privilege masks and status words, this flag controls how the pattern bit mask specified by the item descriptor is compared with that in the process. By default, the bit masks are compared for equality.

The `PSCAN$M_BIT_ANY` flag is used only with bit masks.

PSCAN\$M_CASE_BLIND

When you specify `PSCAN$M_CASE_BLIND` to compare the character string specified by the item descriptor with the character string value from the process, `$PROCESS_SCAN` does not distinguish between uppercase and lowercase letters.

The `PSCAN$M_CASE_BLIND` flag is used only with character-string item codes. The `PSCAN$M_CASE_BLIND` flag can be specified with either the `PSCAN$M_PREFIX_MATCH` flag or the `PSCAN$M_WILDCARD` flag.

PSCAN\$M_EQL

When you specify `PSCAN$M_EQL`, `$PROCESS_SCAN` compares the value specified by the item descriptor with the value from the process to see if there is an exact match.

`PSCAN$M_EQL` and `PSCAN$M_NEQ` are used with bit masks, character strings, and integers to control how the item is interpreted. Only one of the flags can be specified; if more than one of these flags is used the `SS$_IVSSRQ` error is returned. If you want to specify that bits not set in the pattern mask must not be set in the process mask, use `PSCAN$M_EQL`.

PSCAN\$M_GEQ

When you specify `PSCAN$M_GEQ`, `$PROCESS_SCAN` selects a process if the value from the process is greater than or equal to the value specified by the item descriptor.

`PSCAN$M_GEQ`, `PSCAN$M_GTR`, `PSCAN$M_LEQ`, and `PSCAN$M_LSS` are used with integer item codes only. Only one of these four flags can be specified; if more than one of these flags is used the `SS$_IVSSRQ` error is returned.

PSCAN\$M_GTR

When you specify `PSCAN$M_GTR`, `$PROCESS_SCAN` selects a process if the value from the process is greater than the value specified by the item descriptor.

\$PROCESS_SCAN

PSCAN\$M_GEQ, PSCAN\$M_GTR, PSCAN\$M_LEQ, and PSCAN\$M_LSS are used with integer item codes only. Only one of these four flags can be specified; if more than one of these flags is used the SS\$_IVSSRQ error is returned.

PSCAN\$M_LEQ

When you specify PSCAN\$M_LEQ, \$PROCESS_SCAN selects a process if the value from the process is less than or equal to the value specified by the item descriptor.

PSCAN\$M_GEQ, PSCAN\$M_GTR, PSCAN\$M_LEQ, and PSCAN\$M_LSS are used with integer item codes only. Only one of these four flags can be specified; if more than one of these flags is used the SS\$_IVSSRQ error is returned.

PSCAN\$M_LSS

When you specify PSCAN\$M_LSS, \$PROCESS_SCAN selects a process if the value from the process is less than the value specified by the item descriptor.

PSCAN\$M_GEQ, PSCAN\$M_GTR, PSCAN\$M_LEQ, and PSCAN\$M_LSS are used with integer item codes only. Only one of these four flags can be specified; if more than one of these flags is used the SS\$_IVSSRQ error is returned.

PSCAN\$M_NEQ

When you specify PSCAN\$M_NEQ, \$PROCESS_SCAN selects a process if the value from the process is not equal to the value specified by the item descriptor.

PSCAN\$M_EQL and PSCAN\$M_NEQ are used with bit masks, character strings, and integers to control how the item is interpreted. Only one of the flags can be specified; if more than one of these flags is used the SS\$_IVSSRQ error is returned.

PSCAN\$M_OR

When you specify PSCAN\$M_OR, \$PROCESS_SCAN selects processes whose values match the current item descriptor or the next item descriptor. The next item descriptor must have the same item code as the item descriptor with the PSCAN\$M_OR flag. Multiple items are chained together; all except the last item descriptor must have the PSCAN\$M_OR flag.

The PSCAN\$M_OR flag can be specified with any other flag and can be used with bit masks, character strings, and integers. If the PSCAN\$M_OR flag is used between different item codes, or if it is missing between identical item codes, the SS\$_IVSSRQ error is returned.

4-54 System Services Features

\$PROCESS_SCAN

PSCAN\$M_PREFIX_MATCH

When you specify **PSCAN\$M_PREFIX_MATCH**, **\$PROCESS_SCAN** compares the character string specified in the item descriptor to the leading characters of the requested process value.

For example, to find all process names that start with the letters *AB*, use the string *AB* with the **PSCAN\$M_PREFIX_MATCH** flag. If you do not specify the **PSCAN\$M_PREFIX_MATCH** flag, the search looks for a process with the 2-character process name *AB*.

The **PSCAN\$M_PREFIX_MATCH** flag also allows either the **PSCAN\$M_EQL** or the **PSCAN\$M_NEQ** flag to be specified. If you specify **PSCAN\$M_NEQ**, the service matches those names that do *not* begin with the specified character string.

The **PSCAN\$M_PREFIX_MATCH** is used only with character-string item codes. The **PSCAN\$M_PREFIX_MATCH** flag cannot be specified with the **PSCAN\$M_WILDCARD** flag; if both of these flags are used the **SS\$_IVSSRQ** error is returned.

PSCAN\$M_WILDCARD

When you specify **PSCAN\$M_WILDCARD**, the character string specified by the item descriptor is assumed to be a wildcard pattern. Acceptable wildcard characters are the asterisk (***), which allows the match to substitute any number of characters in place of the asterisk, and the percent sign (*%*), which allows the match to substitute any one character in place of the percent sign. For example, if you want to search for all process names that begin with the letter *A* and end with the string *ER*, use the string *A*ER* with the **PSCAN\$M_WILDCARD** flag. If the **PSCAN\$M_WILDCARD** flag is not specified, the search looks for the 4-character process name *A*ER*.

The **PSCAN\$M_WILDCARD** is used only with character-string item codes. The **PSCAN\$M_WILDCARD** flag cannot be specified with the **PSCAN\$M_PREFIX_MATCH** flag; if both of these flags are used the **SS\$_IVSSRQ** error is returned. The **PSCAN\$M_NEQ** flag can be used with **PSCANM\$_WILDCARD** to exclude values during a wildcard search.

description

The Process Scan system service creates and initializes a process context that is used by **\$GETJPI** to scan processes on the local system or across the nodes in a VAXcluster system. An item list is used to specify selection criteria to obtain information about specific processes, for example, all processes owned by one user or all batch processes.

The output of the **\$PROCESS_SCAN** service is a process context longword named **pidctx**. This process context is then provided to **\$GETJPI** as the **pidadr** argument. The process context provided by **\$PROCESS_SCAN**

enables \$GETJPI to search for processes across the nodes in a VAXcluster system and to select processes that match certain selection criteria.

The process context consumes process dynamic memory. This memory is deallocated when the end of the context is reached. For example, when the \$GETJPI service returns SS\$_NOMOREPROC or when \$PROCESS_SCAN is called again with the same **pidctx** longword, the dynamic memory is deallocated. If you anticipate that a scan might be interrupted before it runs out of processes, \$PROCESS_SCAN should be called a second time (without an **itmlst** argument) to release the memory. Dynamic memory is automatically released when the current image terminates.

\$PROCESS_SCAN copies the item list and user buffers to the allocated dynamic memory. This means that the item lists and user buffers can be deallocated or reused immediately; they are not referenced during the calls to \$GETJPI.

The item codes referenced by \$PROCESS_SCAN are found in data structures that are always resident in the system, primarily the process control block (PCB) and the job information block (JIB). A scan of processes never forces a process that is swapped out of memory to be brought into memory to read nonresident information.

condition values returned

SS\$_NORMAL	The service completed successfully.
SS\$_ACCVIO	The pidctx argument cannot be written by the caller, the item list cannot be read by the caller, or a buffer for a reference descriptor cannot be read.
SS\$_BADPARAM	The item list contains an invalid item identifier or an invalid combination of item-specific flags is present.
SS\$_IVBUFLLEN	The buffer length field is invalid. For immediate value descriptors, the buffer length must be zero. For reference descriptors, the buffer length cannot be zero or longer than the maximum for the specified item code. This error is also returned if the total length of the item list plus the length of all of the buffer fields is too large to process.
SS\$_IVSSRQ	The pidctx argument was not supplied, or the item list is improperly formed (for example, multiple occurrences of a given item code were interspersed with other item codes).

Chapter 5

System Error Messages

The following VMS facilities have new or changed error messages for Version 5.2:

- AUDSRV
- CLI
- EXCHANGE/NETWORK
- PRINT
- RMS
- SMI
- SUBMIT
- System
- VAXTPU

Section 5.1 describes new messages. Section 5.2 describes revised messages.

5.1 New Messages

The following messages are new for Version 5.2:

ARCHIVEDSB, archiving disabled

Facility: AUDSRV, Audit Server

Explanation: Archiving is disabled.

User Action: None. This is an informational message.

5-2 System Error Messages

New Messages

ARCHIVEMOVE, archiving ending on: 'filename'; archiving starting on: 'filename'

Facility: AUDSRV, Audit Server

Explanation: Archiving has been successfully redirected to a new archive file.

User Action: None. This is an informational message.

ARCHIVENEW, archiving starting on: 'filename'

Facility: AUDSRV, Audit Server

Explanation: Archiving has been successfully enabled to the specified archive file.

User Action: None. This is an informational message.

ARCNODISK, archive file open failure; archiving is directed to an invalid device type

Facility: AUDSRV, Audit Server

Explanation: Archiving could not be enabled because the archive journal is not directed to a file-structured disk device.

User Action: Redirect the archive journal to a file-structured disk device.

ARCOPENFAIL, archive file open failure (status: 'status')

Facility: AUDSRV, Audit Server

Explanation: The audit server could not open the system audit archive file. The specified status code indicates the exact cause of the problem.

User Action: Correct the indicated error, and try again.

ATPC, at PC = 'xxxxxxx'

Facility: Shared by several facilities

Explanation: This message generally accompanies a message indicating a software failure.

User Action: Take corrective action based on the accompanying messages.

BADDEV, device is unsupported for transfer

Facility: EXCHANGE/NETWORK Command

Explanation: You attempted to transfer the file either to or from a device that is not supported by the device (such as magnetic tape).

User Action: Correct an error in the device specification or try using a different device.

BADLOGIC, internal logic error detected, please report error 'errorcode'

Facility: Shared by several facilities

Explanation: The Exchange Utility or EXCHANGE/NETWORK command encountered an unexpected condition and terminated.

User Action: Please submit a Software Performance Report (SPR), describing the error number and the commands that caused the error message. If the error is reproducible only using a particular piece of media, send a copy of the media with the SPR. (Use the BACKUP/PHYSICAL command to make the copy.)

BADREFCNT, ref count: 'ccc', zap count: 'zzz', address: 'xxxxxxx'

Facility: VAXTPU, VAX Text Processing Utility

Explanation: Internal VAXTPU failure.

User Action: Please submit a Software Performance Report (SPR).

BADTAGVAL, bad boundary tag value

Facility: LIB, Library Facility

Explanation: A call to LIB\$FREE_VM has overwritten memory locations. Tags are longwords appended to the end of each block that store the size of the block. Before a block is freed, LIB\$FREE_VM checks that these blocks have not been overwritten.

User Action: Modify your program to ensure that it writes only to the memory locations that it has access to.

BADWIDTHCHANGE, terminal will not support change of width

Facility: VAXTPU, VAX Text Processing Utility

Explanation: Window widths may not be changed on terminals not made by Digital or on terminals without advanced video.

User Action: Use a different terminal or do not change window widths.

CANNOTUNSEL, cannot unselect item from unselect action routine

Facility: VAXTPU, VAX Text Processing Utility

Explanation: You tried to unselect the global selection within the action routine called when the global selection is being unselected.

User Action: The global item is already unselected; remove the code that attempts to unselect it again.

5-4 System Error Messages

New Messages

CAPTINT, captive account - interactive access denied

Facility: CLI, Command Language Interpreter (DCL)

Explanation: Interactive access (in other words, access to the \$ prompt) is not allowed for captive accounts.

User Action: There is an error in the captive login command procedure that has been established for this account, causing it to end unintentionally. Or, the captive login command procedure has not been set up correctly for this account. See your system manager to have the situation corrected.

CAPTINQ, captive account - inquire command not valid

Facility: CLI, Command Language Interpreter (DCL)

Explanation: The INQUIRE command is not valid for captive accounts.

User Action: See your system manager to have the INQUIRE command removed from the login command procedure and replaced with the READ/PROMPT command.

CLIPBOARDFAIL, unexpected clipboard failure

Facility: VAXTPU, VAX Text Processing Utility

Explanation: An unexpected clipboard failure has occurred.

User Action: A software error has occurred. Please submit a Software Performance Report (SPR).

CLIPBOARDLOCKED, clipboard is locked by another process

Facility: VAXTPU, VAX Text Processing Utility

Explanation: The clipboard is locked by another process.

User Action: Repeat the clipboard operation after the other process has finished with the clipboard.

CLIPBOARDNODATA, clipboard does not contain the requested data

Facility: VAXTPU, VAX Text Processing Utility

Explanation: The clipboard does not contain the requested data.

User Action: Repeat the clipboard operation after the other process has finished with the clipboard.

CLIPBOARDZERO, clipboard data has 0 length

Facility: VAXTPU, VAX Text Processing Utility

Explanation: The data to be written to the clipboard has zero length.

User Action: Specify only nonzero-length data to be written to the clipboard.

CLOSEIN, error closing 'filespec' as input

Facility: EXCHANGE/NETWORK Command

Explanation: The EXCHANGE/NETWORK command encountered an error while closing an input file. This message is usually accompanied by a VMS RMS message indicating the reason for the failure.

User Action: Take corrective action based on the accompanying message.

CLOSEOUT, error closing 'filespec' as output

Facility: EXCHANGE/NETWORK Command

Explanation: The EXCHANGE/NETWORK command encountered an error while closing the output file. This message is usually accompanied by a VMS RMS message indicating the reason for the failure.

User Action: Take corrective action based on the accompanying message.

COPIEDB, 'input-filespec' copied to 'output-filespec' ('nnn' blocks)

Facility: Shared by several facilities

Explanation: The message displays the number of blocks copied, based on the size of the input files.

User Action: None. This message is informational.

COPIEDC, 'input-filespec' copied to 'output-filespec' ('nnn' records packed into 'mmm' blocks)

Facility: Shared by several facilities

Explanation: The message displays the number of records read in from the input file and the number of blocks written to the output file.

User Action: None. This message is informational.

5-6 System Error Messages

New Messages

COPIEDR, 'input-filespec' copied to 'output-filespec' ('nnn' records)

Facility: Shared by several facilities

Explanation: The message displays the number of records copied, based on the size of the input files.

User Action: None. This message is informational.

DISPLAY_SHORT, formatted message exceeds screen length in record 'nnn'

Facility: AUDSRV, Audit Server

Explanation: The entire text of the security alarm message could not be displayed because the terminal page length is not long enough.

User Action: Increase the terminal page length, if possible, and try again.

EXT_ERR, error in RMS Extension; reason code %Xnnnnnnnn

Facility: RMS, VMS Record Management Services

Explanation: An RMS extension discovered an error in processing a file. This most likely indicates inconsistent data in the file, but may indicate that the file has an incorrect STORED_SEMANTICS attribute. The reason code is a VMS status code provided by the RMS extension.

User Action: Use the DIRECTORY/FULL command to check the semantics of the file being accessed. It may be necessary to manually alter the file's semantics attribute using the SET FILE command.

EXTNOTFOU, RMS Extension not found

Facility: RMS, VMS Record Management Services

Explanation: The semantics of the file being accessed did not match the semantics of the attempted access, and no RMS extension could be located that could perform the conversion.

User Action: Use the DIRECTORY/FULL command to check the semantics of the file being accessed. It may be necessary to convert the file (using the DCL CONVERT command) for use by the application that reported the error.

EXTRANEOUSARGS, one or more extraneous arguments specified

Facility: VAXTPU, VAX Text Processing Utility

Explanation: You have specified one or more extraneous arguments for this VMS DECwindows call.

User Action: Specify acceptable arguments only.

FAILURE_STATUS, facility 'name' returned failure status of 'xxxxxxx'

Facility: VAXTPU, VAX Text Processing Utility

Explanation: A facility that VAXTPU called returned a failure status.

User Action: There is an error in the VAXTPU built-in procedure call that generated the error. Correct the parameters and try again.

FDLPARSE, fatal error encountered parsing FDL file

Facility: EXCHANGE/NETWORK Command

Explanation: An error occurred during the parsing of the FDL file supplied by the user. This message is issued with an accompanying message.

User Action: Take corrective action based on the accompanying message and reenter the command.

FRZNCTX, context is frozen - either cancel this context or use alternate

Facility: CLI, Command Language Interpreter (DCL)

Explanation: You are attempting to use the F\$CONTEXT lexical to add context information to a symbol. A context symbol cannot be changed after it has been referenced by an appropriate lexical function other than F\$CONTEXT, which requires a context symbol.

User Action: You must either use another symbol or use the cancel function of F\$CONTEXT to erase the context from this symbol.

GBLSEOWNER, you are the global selection owner

Facility: VAXTPU, VAX Text Processing Utility

Explanation: This VAXTPU session is the global selection owner.

User Action: Because you own the global section, you must process it yourself.

HIGHVER, higher version of 'output-filespec' already exists

Facility: Shared by several facilities

Explanation: An explicit version number is requested for an output file; the directory already contains an entry for the same file name and file type with a higher version number. Note that if the file is subsequently specified in a command, the system will locate the previously existing version if no version number is specified. The newly created output file will not be used.

User Action: None. This message is informational.

5-8 System Error Messages

New Messages

INCONSTATE, internal consistency state; please submit an SPR with
SYS\$SYSROOT:[SYSEXE]AUDIT_SERVER.DMP

Facility: AUDSRV, Audit Server

Explanation: An internal consistency error has occurred in the audit server process.

User Action: Please submit a Software Performance Report (SPR) and include the process dump file SYS\$SYSROOT:[SYSEXE]AUDIT_SERVER.DMP, if one exists.

INVCTXTYP, symbol specified is defined for a different context type

Facility: CLI, Command Language Interpreter (DCL)

Explanation: You must use a symbol that has a type required by the lexical function you are using.

User Action: Refer to the documentation of the lexical function for more information.

INVGBLSELDATA, the selected data cannot be processed

Facility: VAXTPU, VAX Text Processing Utility

Explanation: VAXTPU cannot process the data of the selected region.

User Action: Select 8-bit uniform font text when pasting the global selection into VAXTPU.

INVTIME, invalid time interval

Facility: VAXTPU, VAX Text Processing Utility

Explanation: An invalid time interval was specified.

User Action: Correct the specified time interval.

LISTENDSB, event listening disabled

Facility: AUDSRV, Audit Server

Explanation: Security audit event listening has been disabled.

User Action: None. This is an informational message.

LISTENENAB, event listening enabled on device 'device'

Facility: AUDSRV, Audit Server

Explanation: The specified device has been successfully enabled as a security audit event listener device.

User Action: None. This is an informational message.

LISTENFAIL, device 'device' does not exist or is not accessible; requestor PID: 'pid'

Facility: AUDSRV, Audit Server

Explanation: The specified listener device could not be accessed because the device does not exist or it is not accessible to the audit server process.

User Action: Ensure that the device exists and allows (SYSTEM:RWLP) access.

LISTENNOTMBX, requested listener device 'device' is not a mailbox; requestor PID: 'pid'

Facility: AUDSRV, Audit Server

Explanation: The audit server could not associate the specified listener device because the listener device is not a mailbox device.

User Action: Specify an existing mailbox device.

MODRANGEMARKS, MODIFY_RANGE requires either two marks or none

Facility: VAXTPU, VAX Text Processing Utility

Explanation: You must specify both new ends of the range to be modified or that there are no new ends.

User Action: Correct the call to MODIFY_RANGE, using the BEGINNING_OF or END_OF built-in procedure.

MSGQUEEMPTY, free message queue empty; requestor PID: 'pid'

Facility: AUDSRV, Audit Server

Explanation: The audit server encountered an unexpected internal error.

User Action: Please submit a Software Performance Report (SPR) and include the process dump file SYS\$SYSROOT:[SYSEXE]AUDIT_SERVER.DMP, if one exists.

5-10 System Error Messages

New Messages

NEW_FILE, now analyzing file 'filename'

Facility: AUDSRV, Audit Server

Explanation: A new security audit journal file has been accessed by the Audit Analysis Utility. This message is primarily useful if you have used a wildcard file specification to analyze multiple security audit journal files.

User Action: None. This is an informational message.

NEWIGNORE, security auditing resources ignored; new messages ignored

Facility: AUDSRV, Audit Server

Explanation: The audit server internal message buffer has been exhausted. As a result, the audit server has chosen to ignore a security alarm in an attempt to preserve the events that led up to this point.

User Action: None. This event results from the server FINAL_ACTION setting of IGNORE_NEW.

NEWSERVERDB, new audit server database created

Facility: AUDSRV, Audit Server

Explanation: A new permanent audit server database (SYS\$MANAGER:AUDIT_SERVER.DAT) has been created by the audit server process.

User Action: None. This is an informational message.

NOENTRYSYM, unable to set \$ENTRY symbol to current entry number

Facility: PRINT, PRINT Command and SUBMIT, SUBMIT Command

Explanation: The print or submit command failed to set the global symbol \$ENTRY. However, the actual print or submit request did succeed.

User Action: Take corrective action based on the accompanying messages.

NOCHARREAD, no character was read by the READ_CHAR built-in

Facility: VAXTPU, VAX Text Processing Utility

Explanation: The READ_CHAR built-in procedure did not read a character due to the user pressing a keypad or function key.

User Action: None.

NOFORCMD, account restricted - foreign commands not valid

Facility: CLI, Command Language Interpreter (DCL)

Explanation: Foreign commands have been disabled for this account.

User Action: This account requires that all images are run by verbs defined in the CLI table (DCLTABLES) specified for this account.

NORUNMCR, account restricted - RUN and MCR commands not valid

Facility: CLI, Command Language Interpreter (DCL)

Explanation: The RUN and MCR commands are foreign commands and have been disabled for this account. (See the *VMS DCL Concepts Manual* for an explanation of foreign commands.)

User Action: This account requires that all images are run using verbs defined in the CLI table (DCLTABLES) specified for this account.

NOGBSELDATA, no global selection data

Facility: VAXTPU, VAX Text Processing Utility

Explanation: There is no data in the DECwindows global selection region.

User Action: Select data before attempting to paste the global selection into VAXTPU.

NOGBSELOWNER, there is no global selection owner

Facility: VAXTPU, VAX Text Processing Utility

Explanation: There is no global selection owner.

User Action: Make sure that an application owns the global selection before requesting information about it.

NOOPCOM, OPCOM not running at 'time'; security alarms may be lost

Facility: AUDSRV, Audit Server

Explanation: The audit server detects that the Operator Communication Facility (OPCOM) process is not running.

User Action: From the SYSTEM account, start OPCOM using the DCL command @SYS\$SYSTEM:STARTUP OPCOM.

5-12 System Error Messages

New Messages

NOREDEFINE, built-in procedure 'name' cannot be redefined

Facility: VAXTPU, VAX Text Processing Utility

Explanation: Your program uses a VAXTPU built-in procedure name as the name of a user-written procedure. Built-in procedures cannot be superseded by user-written procedures.

User Action: Give your procedure a name that does not conflict with any names of the VAXTPU built-in procedures.

NORESTART, error 'status' attempting to restart server

Facility: AUDSRV, Audit Server

Explanation: The audit server process encountered an error when attempting to restart itself as a result of a previous error. The displayed status code reflects an error from the \$CREPRC system service.

User Action: Correct the indicated error and restart the audit server manually using the command SET AUDIT/SERVER=START.

NOTCTX, symbol specified is not a valid context symbol

Facility: CLI, Command Language Interpreter (DCL)

Explanation: You are trying to use a symbol that has a type other than a context type such as PROCESS_CONTEXT.

User Action: Refer to the documentation of the lexical function you are trying to use for more information.

NOTIMPLEMENTED, built-in compiled by 'name' is not implemented by 'name'

Facility: VAXTPU, VAX Text Processing Utility

Explanation: The section file contains a reference to a built-in procedure implemented in another version of VAXTPU, but which is not implemented in this version of VAXTPU.

User Action: You must use the version of VAXTPU that compiled the section file.

OBSERVERDB, obsolete audit server database 'n.n' encountered; new database created

Facility: AUDSRV, Audit Server

Explanation: An obsolete audit server database file was encountered by the audit server process during its initialization processing. A new database

was created. The displayed numbers indicate the major and minor database version numbers.

User Action: The site security administrator should ensure that the permanent audit server characteristics are correctly set in the new server database.

OLDPURGE, security auditing resources exhausted; oldest message purged

Facility: AUDSRV, Audit Server

Explanation: The audit server internal message buffer has been exhausted. As a result, the audit server has chosen to purge the oldest security alarm from its internal buffer in an attempt to preserve the newest alarms.

User Action: None. This event results from the server FINAL_ACTION setting of PURGE_OLD.

OPENIN, error opening 'input-filespec' as input

Facility: Shared by several facilities

Explanation: An input file cannot be opened. This message is usually accompanied by a VMS RMS message indicating the reason for the failure.

User Action: Take corrective action based on the accompanying message.

OPENOUT, error opening 'output-filespec' as output

Facility: Shared by several facilities

Explanation: An output file cannot be opened. This message is usually accompanied by a VMS RMS message indicating the reason for the failure.

User Action: Take corrective action based on the accompanying message.

OPNOTSUP, operation not supported by RMS Extension

Facility: RMS, VMS Record Management Services

Explanation: An extended RMS component has been requested to perform a service it is incapable of performing. For example, attempting to do a \$PUT to a file coded with DDIF semantics.

User Action: Use the DIRECTORY/FULL command to check the semantics of the file being accessed. It may be necessary to convert the file (using the DCL command CONVERT) for use by the application that reported the error.

5-14 System Error Messages

New Messages

OVERLAPRANGE, overlapping ranges, operation terminated

Facility: VAXTPU, VAX Text Processing Utility

Explanation: The source and destination ranges of a COPY_TEXT or MOVE_TEXT overlap. The operation was terminated before any modifications were performed.

User Action: Do not specify overlapping ranges to COPY_TEXT or MOVE_TEXT.

PAGLIMEXC, page limit exceeded for zone

Facility: LIB, Library Facility

Explanation: Allocation exceeds the page limit for the zone. A LIB\$GET_VM request has attempted to allocate more pages than specified by the *page_limit* attribute for the zone.

User Action: Modify your program to allocate fewer pages, to create a zone with a higher page limit, or to request pages from another zone.

PEXFULL, process exhaustion list full; requestor PID: 'pid'

Facility: AUDSRV, Audit Server

Explanation: The audit server process exclusion list is full.

User Action: Use the SET AUDIT/NOEXCLUDE command to remove existing entries and try again.

READABORTED, READ_CHAR, READ_KEY, or READ_LINE built-in was aborted

Facility: VAXTPU, VAX Text Processing Utility

Explanation: The READ_CHAR, READ_KEY, or READ_LINE built-in procedure was aborted because the user action required executing other VAXTPU code.

User Action: Return control to VAXTPU so that the other code can execute.

READERR, error reading 'filespec'

Facility: Shared by several facilities

Explanation: An input file specified cannot be read. This message is usually accompanied by a VMS RMS message indicating the reason for the failure.

User Action: Take corrective action based on the accompanying message.

RECPRN, 'filespec' contained 'nnn' records with invalid PRN fields ('mmm' prefix
'ppp' postfix)

Facility: EXCHANGE/NETWORK Command

Explanation: The specified file contained one or more records with invalid printing control fields in the print file control area (PRN). The first byte of the control area constitutes a prefix area, while the second byte constitutes a postfix area. These areas specify the carriage control to be performed before and after printing, respectively. A carriage return and a line-feed character were substituted for each invalid byte, and the file was copied.

User Action: If the substitution is not acceptable, determine the cause of the PRN field error, fix it, and reenter the command.

REGWIDDUP, registration string already associated with a different widget

Facility: VAXTPU, VAX Text Processing Utility

Explanation: The registration string is already associated with a different widget.

User Action: An application is attempting to register multiple widgets with the same name. Modify the application to use unique names for each registered widget.

REMARCEST, remote archive link established ('n' messages lost)

Facility: AUDSRV, Audit Server

Explanation: The audit server has established a new remote archive file connection and enabled archiving to the remote archive file. This message also displays the number of messages lost while the remote node was unavailable.

User Action: None. This is an informational message.

REMARCFAIL, remote archive link failure; archive messages will be lost (status:
'status')

Facility: AUDSRV, Audit Server

Explanation: The network link to a remote system audit archive file has failed.

User Action: None. The audit server will reestablish a link to the remote node as soon as it becomes available.

5-16 System Error Messages

New Messages

REMARCNVL, remote archive link not available; archive messages will be lost

Facility: AUDSRV, Audit Server

Explanation: The audit server could not establish a link to the node associated with the system audit archive file. This message may be accompanied by further information indicating the exact nature of the problem. If no further information is signaled, this indicates that the node is not available or not accepting remote connections.

User Action: None. The audit server will establish a link to the remote node as soon as it becomes available.

REMARCNVLS, remote archive link not available (' n ' messages lost)

Facility: AUDSRV, Audit Server

Explanation: The audit server is unable to establish a link to the node associated with the system audit archive file. This message follows the REMARCNVL or REMARCFAIL error message and is broadcast each minute indicating the number of archive messages being lost.

User Action: None. The audit server will establish a link to the remote node as soon as it becomes available.

REMDISABLED, resource monitoring disabled for journal ' name '

Facility: AUDSRV, Audit Server

Explanation: Resource monitoring has been disabled for the specified security audit journal file.

User Action: None. This is an informational message.

REMENABLED, resource monitoring enabled for journal ' name '

Facility: AUDSRV, Audit Server

Explanation: Resource monitoring has been enabled for the disk volume associated with the specified security audit journal file.

User Action: None. This is an informational message.

REMNOTENAB, resource monitoring not enabled for journal ' name '

Facility: AUDSRV, Audit Server

Explanation: The audit server could not perform the requested action because resource monitoring was not enabled for the specified audit journal file.

User Action: Enable resource monitoring for the specified journal and try again.

REQARGSMISSING, one or more required arguments missing

Facility: VAXTPU, VAX Text Processing Utility

Explanation: You have not specified one or more required arguments for this DECwindows call.

User Action: Specify all required arguments.

REQUIRESDECW, feature requires the VAXTPU DECwindows screen updater

Facility: VAXTPU, VAX Text Processing Utility

Explanation: The requested operator or feature is available only using the VAXTPU DECwindows screen updater.

User Action: Do not use the feature.

REQUIRESTERM, feature requires a terminal

Facility: VAXTPU, VAX Text Processing Utility

Explanation: The requested feature is available only when there is a terminal.

User Action: Either do not use the feature or specify the /DISPLAY qualifier.

RESCRASH, resources exhausted; server restarting

Facility: AUDSRV, Audit Server

Explanation: The process virtual memory allocated to the audit server (used to buffer incoming security alarms) has been exhausted. This message indicates that the audit server final action mode was set to RESTART and that the audit server process is restarting. Buffered security alarms received prior to the resource exhaustion condition will be stored in the process dump file SYS\$SYSROOT:[SYSEXE]AUDIT_SERVER.DMP.

User Action: None. This is an informational message.

RESCRITICAL, security auditing resources exhausted on journal 'name'

Facility: AUDSRV, Audit Server

Explanation: The free disk space on the disk volume associated with the specified audit journal file has fallen below the resource action threshold. An accompanying RESINFO message indicates the number of free disk blocks and the number of disk blocks needed to dismiss the resource warning

5-18 System Error Messages

New Messages

condition. The audit server will suspend noncritical system processes to prevent complete exhaustion of the associated volume.

User Action: Log in to a privileged account and free up disk space on the disk volume associated with the specified audit journal file.

RESDISMISS, resource exhaustion condition dismissed on journal 'name'

Facility: AUDSRV, Audit Server

Explanation: The free disk space on the disk volume associated with the specified audit journal file has risen above the resource warning threshold.

User Action: None. This is an informational message.

RESINFO, resource information: 'n' blocks needed, 'n' blocks available

Facility: AUDSRV, Audit Server

Explanation: This message indicates the number of disk blocks needed to remove the preceding resource exhaustion condition on the volume associated with the specified security audit journal file. The number of disk blocks currently available is also given.

User Action: None. This is an informational message.

RESNOTDISK, resource monitoring ignored for journal 'name'; journal is directed to an invalid device type

Facility: AUDSRV, Audit Server

Explanation: Resource monitoring could not be enabled for the specified audit journal file because the audit journal file is not directed to a file-structured disk device.

User Action: Redirect the audit journal file to a file-structured disk device.

RESOKAY, free resources available on journal 'name'

Facility: AUDSRV, Audit Server

Explanation: The free disk space on the disk volume associated with the specified audit journal file is above all thresholds. An accompanying RESINFO message indicates the number of free disk blocks.

User Action: None.

RESTART, audit server restart requested

Facility: AUDSRV, Audit Server

Explanation: A forced audit server restart was received by the audit server process. The audit server process will restart. Buffered security alarms received prior to the restart are stored in the process dump file `SYS$SYSROOT:[SYSEXE]AUDIT_SERVER.DMP`.

User Action: None. This event results from the server `FINAL_ACTION` setting of `RESTART`.

RESUME, system operation resumed; security auditing resources available

Facility: AUDSRV, Audit Server

Explanation: Normal system operation has been resumed by the audit server process following a resource exhaustion condition.

User Action: None. This is an informational message.

RESUMEFAIL, system operation not resumed; resource condition still exists on journal 'name'

Facility: AUDSRV, Audit Server

Explanation: The audit server was unable to resume normal system activity because the free disk space associated with the specified audit journal file is still less than the action threshold.

User Action: Log in to a privileged account and free up disk space on the disk volume associated with the specified audit journal file.

RESUMEWARN, resource exhaustion condition dismissed (with resources still low) on journal 'name'

Facility: AUDSRV, Audit Server

Explanation: The audit server has resumed system activity following a resource exhaustion condition; however, the free disk space associated with the specified audit journal file is still less than the warning threshold.

User Action: Log in to a privileged account and free up disk space on the disk volume associated with the specified audit journal file.

5-20 System Error Messages

New Messages

RESWARNING, resource warning condition exists on journal 'name'

Facility: AUDSRV, Audit Server

Explanation: The free disk space on the disk volume associated with the specified audit journal file has fallen below the resource warning threshold. An accompanying RESINFO message indicates the number of free disk blocks and the number of disk blocks needed to dismiss the resource warning condition.

User Action: Log in to a privileged account and free up disk space on the disk volume associated with the specified audit journal file.

SEMANTICS, inconsistent usage of RMS Semantics

Facility: RMS, VMS Record Management Services

Explanation: RMS XABITM item codes for stored semantics or access semantics have been used in an inconsistent manner.

User Action: This indicates a coding error. Check the documentation on the use of RMS semantics and verify that the application is calling RMS correctly.

SERVEREXIT, requested audit server shutdown

Facility: AUDSRV, Audit Server

Explanation: The audit server has been shut down.

User Action: None. This is an informational message.

SUSPEND, system operation suspended; security auditing resources exhausted

Facility: AUDSRV, Audit Server

Explanation: The free disk space on the disk volume associated with the specified audit journal file has fallen below the resource action threshold. Consequently, the audit server has suspended all noncritical processes.

User Action: Log in to a privileged account and free up disk space on the disk volume associated with the specified audit journal file.

SYNTAX, error parsing 'string'

Facility: Shared by several facilities

Explanation: The command syntax is invalid. The message displays the rejected portion of the command.

User Action: Use the DCL command HELP or refer to the *VMS DCL Dictionary* for the correct syntax and reenter the command.

SYSJNLFULL, device full error on journal SECURITY; automatic server restart suppressed

Facility: AUDSRV, Audit Server

Explanation: The device associated with the system security audit journal file is full. When this event occurs, the audit server does not restart itself. Buffered security alarms received prior to the error are stored in the process dump file SYS\$SYSROOT:[SYSEXE]AUDIT_SERVER.DMP.

User Action: Free up space on the system volume and restart the audit server process using the command SET AUDIT/SERVER=START.

SYSJNLNAC, system audit journal inaccessible; SYS\$COMMON:[SYSMGR]SECURITY_AUDIT.AUDIT\$JOURNAL used instead

Facility: AUDSRV, Audit Server

Explanation: The audit server process cannot open the system security audit log file. This condition occurs, for example, when the device on which the log file resides is unavailable or full. Audit messages are redirected to the default system security audit log file, SECURITY_AUDIT.AUDIT\$JOURNAL, in the SYS\$MANAGER directory.

User Action: Correct the device problem and redirect the system security audit log file to the alternate device.

TIMEOUT, built-in timed out

Facility: VAXTPU, VAX Text Processing Utility

Explanation: The built-in procedure timed out.

User Action: A software error has occurred. Please submit a Software Performance Report (SPR).

UNDWIDCLA, undefined widget class specified

Facility: VAXTPU, VAX Text Processing Utility

Explanation: A widget class integer that is not known to VAXTPU was specified to the CREATE_WIDGET built-in procedure.

User Action: Define a widget class using the DEFINE_WIDGET_CLASS built-in procedure before attempting to call CREATE_WIDGET.

5-22 System Error Messages

New Messages

WIDMISMATCH, parameter 'numbers' class, 'class', unsupported

Facility: VAXTPU, VAX Text Processing Utility

Explanation: The wrong type of widget has been passed to a VAXTPU built-in procedure. Usually, the procedure can accept a simple text widget only.

User Action: Correct your code and try again.

WRITEERR, error writing 'filespec'

Facility: Shared by several facilities

Explanation: The output file cannot be written. This message should be accompanied by a VMS RMS message indicating the reason for the error.

User Action: Follow the recovery procedure for the specified VMS RMS message.

5.2 Changes to Existing Messages

The following messages have been changed:

SYSVERDIF, system version mismatch - please relink

Facility: RMS, VMS Record Management Services

Explanation: A SYS.STB that links the image being run is different from that of the currently running system.

User Action: Relink the image.

FILEGONE, startup component database is missing

Facility: SMI, System Management Integrator/Server

Explanation: The current startup component database does not exist, or you do not have read or write access to this database.

User Action: To create a new database, use the STARTUP CREATE command at the SYSMAN> prompt. To select a different component database, use the STARTUP SET DATABASE command at the SYSMAN> prompt.

If the database exists, but you cannot access it, verify the protection applied to the file (using the DIRECTORY/FULL command) and change the protection if necessary (using the SET PROTECTION command). If you cannot change the protection, ask the system manager or the owner of the file to change the protection.

Appendix A

Version 5.1 Features

This appendix includes information from the *VMS Version 5.1 New Features Manual* that has not yet been incorporated into Version 5.2 documentation.

A.1 Support for Compound Documents

The term **compound documents** refers to files that may contain a number of integrated components including text, graphics, and scanned images. This appendix specifically describes VMS support for using the text from DECwindows compound documents that are structured according to the Digital Document Interchange Format (DDIF) specification. Refer to the *VMS Compound Document Architecture Manual* for more information about compound documents.

VMS commands and utilities, as well as existing application programs that accept text input, can now use the text content of DECwindows compound documents.

To support the use of DDIF text, VMS RMS has implemented a new RMS file attribute, **stored semantics**, and a DDIF-to-text **RMS extension**. The value of the stored semantics attribute is called the file **tag** and it specifies how file data is to be interpreted. When file data is to be interpreted in accordance with the DDIF specification, the appropriate file tag is DDIF. The use of file tags is limited to disk files on VMS Version 5.1 and later systems.

The DDIF-to-text RMS extension transparently extracts text from DDIF files as variable-length text records that can be accessed through the VMS RMS interface.

The enhancements made to support the reading of text from DDIF files are transparent to the user and to the application programmer. This support requires that all DDIF files in a VMS Version 5.1 environment be tagged with the DDIF file tag. DDIF files created by VMS and VMS layered products are tagged appropriately.

Section A.1.1 describes various VMS file management commands and utilities that display, create, and preserve file tags where appropriate. Section A.1.1 also describes the way various VMS commands and utilities respond to DDIF file input. Section A.1.2 describes VMS support for DDIF files in heterogeneous computing environments. Section A.1.3 describes the changes made to the VMS

A-2 Version 5.1 Features

Support for Compound Documents

RMS program interface to support the stored semantics attribute and to control access to the content of DDIF files.

A.1.1 VMS Commands and Utilities

This section describes the VMS commands and utilities that support tag maintenance by displaying, creating, and preserving the RMS file tags used with DDIF files. It also provides additional information that is relevant to the way selected VMS commands and utilities respond to DDIF file input.

The following table lists the VMS commands and utilities that support tag maintenance:

Command/Utility	Tag Maintenance Function
DIRECTORY/FULL	Displays file tag
ANALYZE/RMS_FILE	Displays file tag
SET FILE/SEMANTICS	Creates file tag
VMS MAIL	Preserves file tag†
COPY	Preserves file tag†
BACKUP	Preserves file tag

†See text for exceptions.

Tags are made up of binary values that can be up to 64 bytes long and can be expressed using hexadecimal notation. The hexadecimal value of the DDIF tag, for example, is 2B0C8773010301. VMS permits you to assign mnemonics to tag values so that DCL commands such as DIRECTORY/FULL and VMS utilities such as FDL and ANALYZE/RMS_FILE display a mnemonic for the DDIF tag instead of the hexadecimal value. The following DCL commands have been included in the system startup command file to assign the mnemonic DDIF to the hexadecimal value for a DDIF tag:

```
$ DEFINE/TABLE=RMS$SEMANTIC_TAGS DDIF 2B0C8773010301
$ DEFINE/TABLE=RMS$SEMANTIC_OBJECTS 2B0C8773010301 DDIF
```

Using the appropriate DEFINE commands, you can assign mnemonics for other tags, including tags used with international program applications.

A.1.1.1 Displaying RMS File Tags

The DIRECTORY/FULL command and the Analyze/RMS_File Utility now display the RMS file tag for DDIF files.

A.1.1.1.1 DIRECTORY/FULL

Where applicable, the **DIRECTORY/FULL** command now provides the value of the stored semantics tag as part of the file information returned to the user. This is the recommended method for quickly determining whether or not a file is tagged. The following display illustrates how the **DIRECTORY/FULL** command returns the RMS attributes for a DDIF file named X.DDIF:

```
X.DDIF;1                               File ID:  (767,20658,0)
.
.
.
RMS attributes:      Stored semantics: DDIF
.
.
.
```

A.1.1.1.2 ANALYZE/RMS_FILE

When you use the **ANALYZE/RMS_FILE** command to analyze a DDIF file, the utility returns the file tag as an RMS file attribute.

```
FILE HEADER
File Spec: USERD$:[TEST]X.DDIF;1
.
.
.
Stored semantics: DDIF
.
.
.
```

One **ANALYZE/RMS_FILE** command option is to create an output FDL file that reflects the results of the analysis.

```
$ ANALYZE/RMS_FILE/FDL filespec
```

When you use this option for analyzing a tagged file, the output FDL file includes the file tag as a secondary attribute to the **FILE** primary attribute. This is illustrated in the following FDL file excerpt:

```
IDENT      " 9-JUN-1989 13:27:30  VAX/VMS ANALYZE/RMS_FILE Utility"
.
.
.
SYSTEM
SOURCE                VMS
FILE
ALLOCATION              3
.
.
.
STORED_SEMANTICS      %X'2B0C8773010301' ! DDIF
.
.
.
```

A-4 Version 5.1 Features

Support for Compound Documents

A.1.1.2 Creating RMS File Tags

The CDA\$CREATE_FILE routine in the Compound Document Architecture toolkit creates and tags DDIF files. However, you might encounter a DDIF file that was created without a file tag or a DDIF file whose file tag was not preserved during file processing.

The DCL command SET FILE provides a qualifier, /[NO]SEMANTICS, that permits you to tag a DDIF file through the DCL interface for VMS Version 5.1 or later systems. You can also use the qualifier to change a tag or to remove a tag from a file.

The following command line tags the file X.DDIF as a DDIF file by assigning the appropriate value to the /SEMANTICS qualifier:

```
$ SET FILE X.DDIF/SEMANTICS=DDIF
```

See Section A.1.1 for information about how to use logical name tables to assign a mnemonic to a tag.

A subsequent DIRECTORY/FULL command displays the following line as part of the file header:

```
.  
.  
.  
RMS attributes:      Stored semantics: DDIF  
.  
.  
.
```

The next example illustrates how to use the SET FILE command to delete an RMS file tag:

```
$ SET FILE X.DDIF/NOSEMANTICS
```

A.1.1.3 Preserving RMS File Tags and DDIF Semantics

The COPY command and the VMS Mail Utility preserve RMS file tags and DDIF semantics when you copy or mail a DDIF file on a VMS Version 5.1 or later system, except for conditions described in Sections A.1.2.2, A.1.2.3, and A.1.2.4.

The Backup Utility always preserves file tags and semantics when you back up a DDIF file to magnetic tape.

A.1.1.3.1 COPY Command

This section describes the results of using the COPY command with DDIF files for various operations.

When you copy a DDIF file to a disk on a VMS Version 5.1 or later system using the COPY command, VMS RMS preserves the DDIF tag and the DDIF semantics of the input file in the output file.

When you copy a DDIF file to a nondisk device on a VMS Version 5.1 or later system using the COPY command, VMS RMS does *not* preserve the DDIF tag or the DDIF semantics of the input file in the output file. Instead, VMS RMS writes the text from the input file to the output file as variable-length records.

When you copy two or more DDIF and text files in any combination to a single output file, the output file takes the characteristics of the first input file, as shown in the following examples:

1. In this example, the first input file is a text file, so the output file (FOO.TXT) contains variable-length text records from X.TXT, Y.DDIF, and Z.TXT, but does not include the DDIF tag from Y.DDIF.

```
$ COPY X.TXT,Y.DDIF,Z.TXT FOO.TXT
```

2. In this example, the first input file (A.DDIF) is a DDIF file, so the output file (FOO.DDIF) includes the DDIF tag as well as the DDIF semantics from A.DDIF. The attempt to copy the text input file (Z.TXT) fails because there is no text-to-DDIF RMS extension, but the contents of B.DDIF and C.DDIF are copied to the output file. However, the output file has no practical use because, as a result of the way DDIF files are structured, only the data from the first input file (A.DDIF) is accessible in the output file.

```
$ COPY A.DDIF,B.DDIF,Z.TXT,C.DDIF FOO.DDIF
```

3. In this example, the first input file (A.DDIF) is a DDIF file, so the output file (FOO.DDIF) includes the DDIF tag as well as the contents of A.DDIF. FOO.DDIF also includes the contents of B.DDIF and C.DDIF. Again, however, the output file has no practical use because, as a result of the way DDIF files are structured, only the data from the first input file (A.DDIF) is accessible in the output file.

```
$ COPY A.DDIF,B.DDIF,C.DDIF FOO.DDIF
```

A.1.1.3.2 VMS Mail Utility

The VMS Mail Utility preserves the DDIF file tag when DDIF files are mailed between VMS Version 5.1 or later systems. The VMS Mail Utility also preserves the DDIF file tag when you create an output file on a VMS Version 5.1 or later system using the EXTRACT command.

When you read a mail message that is a DDIF file, the VMS Mail Utility outputs only the text portion of the file. Similarly, if you edit a DDIF mail file, you can access only the file text; the output file is a text file that can no longer be used as a DDIF file. However, if you forward a message that consists of a DDIF file, the VMS Mail Utility sends the entire DDIF file, including DDIF semantics and the DDIF tag, to the addressee.

A-6 Version 5.1 Features

Support for Compound Documents

A.1.1.4 APPEND Command

This section describes what happens when you attempt to use the APPEND command with DDIF and text files.

In the first example, the APPEND command appends a DDIF file to a text file:

```
$ APPEND X.DDIF Y.TXT
```

The output file, Y.TXT, contains its original text records as well as text from the input file, X.DDIF, reformatted as variable-length text records.

In the next example, the APPEND command appends a DDIF file to another DDIF file:

```
$ APPEND X.DDIF Y.DDIF
```

The output file, Y.DDIF, contains the DDIF tag, the original contents of Y.DDIF, and the contents of X.DDIF. However, the portion of the file that contains X.DDIF is not accessible because of the way DDIF files are structured.

In the final example, the APPEND command attempts to append a text file to a DDIF file:

```
$ APPEND X.TXT Y.DDIF
```

This append operation fails because there is no text-to-DDIF RMS extension.

A.1.2 DDIF Support in a Heterogeneous Environment

This section describes the implementation of DDIF support in two heterogeneous environments. The first heterogeneous environment includes VMS Version 5.1 or later systems and non-VMS systems. The second heterogeneous environment includes VMS Version 5.1 or earlier systems.

A.1.2.1 EXCHANGE/NETWORK Command

A new DCL command, EXCHANGE/NETWORK, has been created to support the transfer of files between VMS systems and non-VMS systems that do not support VMS file types. The EXCHANGE/NETWORK command transfers files in either record mode or block mode but can be used only when both systems support DECnet file transfers.

To interactively tag a DDIF file and transfer the file between a non-VMS operating system and a VMS Version 5.1 or later system, do the following:

1. Create the following file, assigning it the name DDIF.FDL:

```
FILE
    ORGANIZATION          sequential
    STORED_SEMANTICS      DDIF

RECORD
    CARRIAGE_CONTROL      none
    FORMAT                 fixed
    SIZE                   512
```


2. Use the following DCL command to transfer the desired file:

```
EXCHANGE/NETWORK/FDL=DDIF.FDL input_filespec output_filespec
```

See Section A.2 for more information about the EXCHANGE/NETWORK command.

A.1.2.2 COPY Command

If you use the COPY command to copy tagged DDIF files to systems other than VMS Version 5.1 systems from a VMS Version 5.1 system, the results will vary depending on the target system:

- If the target system is a non-VMS system, the file is copied, but the DDIF tag is not preserved.
- If the target system is a VMS Version 5.1 or earlier system, the copy operation fails with the VMS RMS error message RMS\$_SUPPORT, network operation not supported, and a secondary error message of RMS\$_SEMANTICS, inconsistent usage of RMS Semantics. Error messages similar to the following will appear:

```
%COPY-E-OPENOUT, error opening PWEDGE::[]TRY.DDIF;1 as output  
-RMS-F-SUPPORT, network operation not supported  
-RMS-E-SEMANTICS, inconsistent usage of RMS Semantics  
%COPY-W-NOTCOPIED, ABCD4:[DAVIDS]TRY.DDIF;1 not copied
```

- If the target system is a cluster alias for a mixed version cluster containing Version 5.1 or earlier systems, the result of the copy operation depends on whether the cluster node that actually handles the request is a Version 5.1 or earlier system.
- If you use the COPY command to copy tagged DDIF files from Version 5.1 or later systems to earlier systems while on an earlier system, the copy operation will fail with the error message RMS\$_NET, network operation failed at remote node, and with a DAP status code of 16F, inconsistent usage of RMS Semantics. Error messages similar to the following will appear:

```
%COPY-E-OPENIN, error opening ARC"davids password"::ABCD4:[DAVIDS]TRY.DDIF;1 as  
input  
-RMS-F-NET, network operation failed at remote node; DAP code = 01F7516F  
%COPY-W-NOTCOPIED, ARC"davids password"::ABCD4:[DAVIDS]TRY.DDIF;1 not copied  
PWEDGE$
```

A.1.2.3 VMS Mail Utility

If you try to send mail messages containing DDIF files to non-VMS systems that do not support tagged files, the VMS Mail Utility returns the NOACCEPTMSG error message, indicating that the remote node cannot accept the message format.

Similarly, the VMS Mail Utility does not support the mailing of DDIF files to systems earlier than Version 5.1. As with non-VMS systems, the VMS Mail Utility returns the NOACCEPTMSG error message for systems earlier than Version 5.1, indicating that the remote node cannot accept the message format.

A-8 Version 5.1 Features

Support for Compound Documents

A.1.2.4 DDIF File Access Within a Mixed Version Cluster

In a cluster that contains both Version 5.1 or earlier systems, operations on DDIF files from systems earlier than Version 5.1 will cause inconsistent behavior. Records read from DDIF files on systems earlier than Version 5.1 will be fixed-length 512-byte records, which contain DDIF control information in addition to the text context. Thus, typing a DDIF file on a system earlier than Version 5.1 does not produce readable text.

Copying a DDIF file using a system earlier than Version 5.1 will not preserve the DDIF tag on the output file, which will cause problems in later access to the new file from a Version 5.1 or later system.

However, using the Backup Utility from systems earlier than Version 5.1 will create a correct backup of DDIF files, and will properly restore DDIF files from BACKUP save sets.

A.1.3 VMS RMS Interface Changes

This section provides details about the changes made to the VMS RMS interface that support access to text in VMS DECwindows DDIF files. It includes information related to tagging files and accessing tagged files through the VMS RMS interface. The section also describes how tags are preserved at the VMS RMS interface.

A.1.3.1 Programming Interface for File Tagging

This section focuses on the use of the DDIF tag for supporting VMS DECwindows files, although VMS RMS also supports file tagging for other compound document data formats.

You can tag a file from the VMS RMS interface by using the \$CREATE service in conjunction with a new extended attribute block (XAB) called the item XAB (\$XABITM). The \$XABITM macro is a general-purpose macro that was added to the RMS interface to support several Version 5.0 features. Tagged file support involves the use of the two item codes shown in Table A-1.

Table A-1: Tag Support Item Codes

Item	Buffer Size	Function
XAB\$_STORED_SEMANTICS	64 bytes maximum	Defines the file semantics established when the file is created
XAB\$_ACCESS_SEMANTICS	64 bytes maximum	Defines the file semantics desired by the accessing program

The entries XAB\$_STORED_SEMANTICS and XAB\$_ACCESS_SEMANTICS in the item list can represent either a control (set) function or a monitor (sense) function that can be passed to VMS RMS from the application program by way of the RMS interface.

The symbolic value `XAB$K_SEMANTICS_MAX_LEN` represents the tag length. This value may be used to allocate buffer space for sensing and setting stored semantics for the DDIF file.

Within any one `$XABITM`, you can activate either the set function or the sense function for the `XAB$_STORED_SEMANTICS` and `XAB$_ACCESS_SEMANTICS` items, because a common field (`XAB$_MODE`) determines which function is active. If you want to activate both the set function and the sense function for either or both items, you must use two `$XABITM` control blocks, one for setting the functions and one for sensing the functions.

Each entry in the item list addressed by the `$XABITM` is made up of three longwords and a longword of zero terminates the list. You can locate the item list anywhere within the readable address space for a process, but any buffers required by the related function must be located in read/write memory. If the item list is invalid, RMS returns a status of `RMS$_XAB` in the `RAB$_L_STS` field and the address of the XAB in `RAB$_L_STV`.

The format and arguments of the `$XABITM` macro are as follows. Note that the block length field and the type code field are statically initialized by the `$XABITM` macro or may be explicitly initialized using a high-level language.

\$XABITM

format

```
$XABITM  ITEMLIST=item-list-address,  
          MODE= { sensemode },  
              { setmode },  
          NXT=next-xab-address
```

arguments

The ITEMLIST argument defaults to zero, but a valid pointer must be specified when you use a XABITM. MODE defaults to *sensemode*. The symbolic offset, size, and a brief description of each XABITM field are described in the following list:

- The block length field (XAB\$B_BLN) is a 1-byte static field that defines the length of the XABITM, in bytes. This field is initialized to the value XAB\$C_ITMLEN.
- The type code (XAB\$B_COD) field is a 1-byte static field that identifies this control block as a XABITM. This field is initialized to the value XAB\$C_ITM.
- The XAB\$L_ITEMLIST field is a longword field that contains the symbolic address of the item list.
- The XAB\$B_MODE field is a 1-byte field that specifies whether the items can be set by the program. It contains either the symbolic value XAB\$K_SETMODE or the symbolic value XAB\$K_SENSEMODE (default).
- The XAB\$L_NXT field is a longword field that contains the symbolic address of the next XAB in the XAB chain. A value of zero (the default) indicates that the current XAB is the last (or only) XAB in the chain.

Example A-1 illustrates a BLISS-32 program that tags a file through the RMS interface. The tag value shown is a 6-byte hexadecimal number representing the code for the DDIF tag. The VMS RMS program interface accepts only hexadecimal tag values.

To write to a tagged file without using an RMS extension, the application program must specify access semantics that match the file's stored semantics. As shown in the example, the \$CREATE service tags the file and the \$CONNECT service specifies the appropriate access semantics.

Example A-1: Tagging a File

```

MODULE TYPE$MAIN (
    IDENT = 'X-1',
    MAIN = MAIN,
    ADDRESSING_MODE (EXTERNAL=GENERAL)
) =
BEGIN
!
! FORWARD ROUTINE
    MAIN : NOVALUE;                ! Main routine
!
! INCLUDE FILES:
!
LIBRARY 'SYS$LIBRARY:LIB';
OWN
    NAM          : $NAM(),
    RETLEN,
    DDIF_TAG     : BLOCK[ 7, BYTE]
                INITIAL( BYTE(%X'2B', %X'0C', %X'87', %X'73', %X'01', %X'03', %X'01')),
    FAB_XABITM   :
                $xabitm
                ( itemlist=
                  $ITMLST_UPLIT
                  (
                    (ITMCOB=XAB$_STORED_SEMANTICS,
                     BUFADR=DDIF_TAG,
                     BUFSIZ=%ALLOCATION(DDIF_TAG))
                  ),
                  mode = SETMODE),
    RAB_XABITM   :
                $xabitm
                ( itemlist=
                  $ITMLST_UPLIT
                  (
                    (ITMCOB=XAB$_ACCESS_SEMANTICS,
                     BUFADR=DDIF_TAG,
                     BUFSIZ=%ALLOCATION(DDIF_TAG))
                  ),
                  mode = SETMODE),
    FAB          : $FAB( fnm = 'TAGGED-FILE.TEST',
                        nam = NAM,
                        mrs = 512,

```

A-12 Version 5.1 Features

Support for Compound Documents

Example A-1 (Cont.): Tagging a File

```

                                rfm = FIX,
                                fac = <GET,PUT,UPD>,
                                xab = FAB_XABITM),
REC                               : BLOCK[512,BYTE],
STATUS,
RAB                               : $RAB( xab = RAB_XABITM,
                                fab = FAB,
                                rsz = 512,
                                rbf = REC,
                                usz = 512,
                                ubf = REC),
DESC                               : BLOCK[8,BYTE] INITIAL(0);
ROUTINE MAIN : NOVALUE =
BEGIN
STATUS = $CREATE( FAB = FAB );
IF NOT .STATUS
THEN
    SIGNAL (.STATUS);
STATUS = $CONNECT( RAB = RAB );
IF NOT .STATUS
THEN
    SIGNAL (.STATUS);
STATUS = $CLOSE( FAB = FAB );
IF NOT .STATUS
THEN
    SIGNAL (.STATUS);
END;
END
ELUDOM
```

A.1.3.2 Accessing a Tagged File

This section provides details of how VMS RMS handles access to tagged files at the program level. When a program accesses a tagged file, VMS RMS must determine whether and when to associate an RMS extension with the access. This is important to the programmer because an RMS extension may change the attributes of the accessed file.

For example, a DDIF file is stored as a sequentially organized file having 512-byte, fixed-length records. If the DDIF-to-text RMS extension is used to extract text from a DDIF file, the accessed file appears as a sequentially organized file having variable-length records with a maximum record size of 2048 bytes and an implicit carriage return.

One consideration in determining whether an access requires the RMS extension is the type of access (FAB\$B_FAC). When an application program opens a file through the VMS RMS program interface, it must specify if it will be doing record I/O (default), block I/O (BIO), or mixed I/O (BRO), where the program has the option of using either block I/O or record I/O for each access. For example, if block I/O operations are specified, VMS RMS does not associate the RMS extension with the file access.

Another consideration is whether the program senses the tag when it opens a file. If the program does not sense the tag when it opens a DDIF file for record access, VMS RMS associates the RMS extension during the \$OPEN and returns the file attributes that have been modified by the extension.

The final consideration is the access semantics the program specifies and the file's stored semantics (tag). If the program specifies block I/O (FAB\$V_BIO) operations, RMS does not associate the RMS extension and the \$OPEN service returns the file's stored attributes to the accessing program regardless of whether the program senses tags.

A.1.3.2.1 File Accesses That Do Not Sense Tags

This section describes what happens when a program does not use the XABITM to sense a tag when it opens a file.

When a program opens a DDIF file for record operations and does not sense the tag, VMS RMS assumes that the program wants to access text in the file. In this case, VMS RMS associates the RMS extension, which provides file attributes that correspond to record-mode access.

When a program opens a DDIF file with the FAB\$V_BRO option and does not sense the tag, any subsequent attempt to use block I/O fails. If the program specifies block I/O (FAB\$V_BIO) when it invokes the \$CONNECT service, the operation fails because the file attributes returned at \$OPEN permit record access only. Similarly, if the program specifies the FAB\$V_BRO option when it opens the file, and then specifies mixed mode (block/record) operations by not specifying RAB\$V_BIO at \$CONNECT time, block operations such as READ and WRITE are disallowed.

A.1.3.2.2 File Accesses That Sense Tags

VMS RMS does not associate the RMS extension as part of the \$OPEN service if a program opens a DDIF file and senses the stored semantics. This allows the program to specify access semantics with the \$CONNECT service. VMS RMS returns the file attributes, including the stored semantics attribute (tag value), to the program as part of the \$OPEN service.

When the program subsequently invokes the \$CONNECT service, VMS RMS uses the specified operations mode to determine its response. If the program specified FAB\$V_BRO with the \$OPEN service and then specifies block I/O (RAB\$V_BIO) when it invokes the \$CONNECT service, VMS RMS does not associate the RMS extension.

But if the program specifies record access or FAB\$V_BRO when it opens the file and then decides to use record I/O when it invokes the \$CONNECT service, VMS RMS compares the access semantics with the file's stored semantics to determine whether to associate the RMS extension. If the access semantics match the stored semantics, VMS RMS does not associate the RMS extension. If the access semantics do not match the stored semantics, VMS RMS associates the access with the RMS extension. In this case, the program must use the

A-14 Version 5.1 Features

Support for Compound Documents

\$DISPLAY service to obtain the modified file attributes. If VMS RMS cannot find the appropriate RMS extension, the operation fails and the \$CONNECT service returns the EXTNOTFOU error message.

If the application program senses the file's stored semantics, VMS RMS allows mixed-mode operations. In this case, mixed block and record operations are permitted because the application gets record mode file attributes and data from the RMS extension and block mode file attributes and data from the file.

Example A-2 illustrates a BLISS-32 program that accesses a tagged file from an application program that does not use an RMS extension.

Example A-2: Accessing a Tagged File

```
MODULE TYPE$MAIN (
    IDENT = 'X-1',
    MAIN = MAIN,
    ADDRESSING_MODE (EXTERNAL=GENERAL)
) =
BEGIN
!
! FORWARD ROUTINE
    MAIN : NOVALUE;                                ! Main routine
!
! INCLUDE FILES:
!
LIBRARY 'SYS$LIBRARY:STARLET';
OWN
    NAM                : $NAM(),
    ITEM_BUFF          : BLOCK[ XAB$K_SEMANTICS_MAX_LEN, BYTE ],
    RETLEN,
    FAB_XABITM        :
        $xabitm
            ( itemlist=
                $ITMLST_UPLIT
                    ((ITM$COD=XAB$ STORED_SEMANTICS,
                        BUFADR=ITEM_BUFF,
                        BUFSIZ=XAB$K_SEMANTICS_MAX_LEN,
                        RETLEN=RETLEN)),
                    mode = SENSEMODE),
    RAB_ITEMLIST       : BLOCK[ ITM$S_ITEM + 4, BYTE ],
    RAB_XABITM         : $XABITM
            ( itemlist=RAB_ITEMLIST,
              mode=SETMODE ),
    FAB                : $FAB( fnm = 'TAGGED-FILE.TEST',
                                nam = NAM,
                                fac = <GET,PUT,UPD>,
                                xab = FAB_XABITM),
    REC                : BLOCK[512,BYTE],
    STATUS,
    RAB                : $RAB( xab = RAB_XABITM,
                                fab = FAB,
```

(continued on next page)

Example A-2 (Cont.): Accessing a Tagged File

```

                                rsz = 512,
                                rbf = REC,
                                usz = 512,
                                ubf = REC),
DESC                            : BLOCK[8,BYTE] INITIAL(0);
ROUTINE MAIN : NOVALUE =
BEGIN
STATUS = $OPEN( FAB = FAB );
IF NOT .STATUS
THEN
    SIGNAL (.STATUS);
RAB_ITEMLIST[ ITM$W_BUFSIZ ] = .RETLEN;
RAB_ITEMLIST[ ITM$L_BUFADR ] = ITEM_BUFF;
RAB_ITEMLIST[ ITM$W_ITMCO ] = XAB$ ACCESS_SEMANTICS;
STATUS = $CONNECT( RAB = RAB );
IF NOT .STATUS
THEN
    SIGNAL (.STATUS);
STATUS = $CLOSE( FAB = FAB );
IF NOT .STATUS
THEN
    SIGNAL (.STATUS);
END;
END
ELUDOM

```

A.1.3.3 Preserving Tags

To preserve the integrity of a tagged file that is being copied or transmitted, the tag must be preserved in the destination (output) file. The most efficient way to use the RMS interface for propagating tags is to open the source file (input) and sense the tag using a \$XABITM with the item code XAB\$_STORED_SEMANTICS:

```

.
.
.
ITEMLIST[ ITM$W_BUFSIZ ] = XAB$K_SEMANTICS_MAX_LEN;
ITEMLIST[ ITM$L_BUFADR ] = ITEM_BUFF;
ITEMLIST[ ITM$L_RETLEN ] = RETLEN;
ITEMLIST[ ITM$W_ITMCO ] = XAB$_STORED_SEMANTICS;
.
.
.
XABITM[ XAB$B_MODE ] = XAB$K_SENSEMODE;
STATUS = $OPEN( FAB = FAB );
.
.
.

```

Then create the destination (output) file and set the tag using a \$XABITM with the item code XAB\$_STORED_SEMANTICS:

A-16 Version 5.1 Features

Support for Compound Documents

```
.  
. .  
IF .RETLEN GTR 0  
THEN  
  BEGIN  
    ITEMLIST[ ITM$W_ITMCO ] = XAB$_STORED_SEMANTICS;  
    ITEMLIST[ ITM$L_SIZE   ] = .RETLEN;  
    XABITM[ XAB$_MODE ] = XAB$_SETMODE;  
  END;  
  
STATUS = $CREATE( FAB = FAB );  
  
. .  
END;  
END  
ELUDOM
```

A.1.4 Distributed File System Support for DDIF Tagged Files

Version 1.1 of the Distributed File System (DFS) includes limited support for DDIF tagged files. You can create and read DDIF files on a DFS device when the DFS client node is running VMS Version 5.1 or later versions. You can also use the DIRECTORY/FULL command to determine whether a DDIF file on a DFS device is tagged.

You cannot use the SET FILE/[NO]SEMANTICS command either to tag DDIF files or to remove the tags from DDIF files on a DFS device. Furthermore, the Backup Utility does not preserve the DDIF tag or the DDIF stored semantics for data files on a DFS device.

A.1.5 VMS RMS Errors

Four VMS RMS error messages signal the user when the appropriate error condition exists:

- RMS\$_EXTNOTFOU
- RMS\$_SEMANTICS
- RMS\$_EXT_ERR
- RMS\$_OPNOTSUP

The RMS\$_EXTNOTFOU error message indicates that VMS RMS has not found the specified RMS extension. Verify that the file is correctly tagged, using the DIRECTORY/FULL command, and that the application program is specifying the appropriate access semantics.

VMS RMS returns the RMS\$_SEMANTICS error message when you try to create a tagged file on a remote system earlier than VMS Version 5.1 from a Version 5.1 or later system.

VMS RMS returns the RMS\$_EXT_ERR error when the DDIF RMS extension detects an inconsistency.

VMS RMS returns the RMS\$_OPNOTSUP error when the RMS DDIF extension is invoked by an RMS operation. For example, if the extension does not support write access to a DDIF file, verify that the application program is not performing record operations that modify the file.

A.2 EXCHANGE/NETWORK Command

The following sections describe the DCL command EXCHANGE/NETWORK.

EXCHANGE/NETWORK

The EXCHANGE/NETWORK command allows the VMS operating system to transfer files to or from operating systems that do not support VMS file organizations. The transfer occurs over a DECnet network communications link that connects VMS and non-VMS operating system nodes.

Using DECnet services, the EXCHANGE/NETWORK command can perform the following operations:

- Transfer files between a VMS node and a non-VMS system node
- Transfer a group of input files to a group of output files
- Transfer files between two non-VMS nodes, provided those nodes share DECnet connections with the VMS node that issues the EXCHANGE/NETWORK command

The EXCHANGE/NETWORK command imposes the following restrictions:

- Transfers of files can occur only between disk devices. (If a disk device is not the desired permanent residence for the file, you must either move the file to a disk before issuing the command or retrieve the file from a disk after the command completes.)
- The remote system must have a block size of 512 bytes, where a byte is 8 bits long.
- The nodes transferring files must support the DECnet Data Access Protocol (DAP).

The VMS Record Management Services (RMS) facility provides VMS access to records in VMS RMS files. To transfer VMS RMS files between two nodes where both nodes are VMS nodes, use one of the other DCL commands (such as COPY, APPEND, or CONVERT), as appropriate. These commands recognize RMS file organizations and are designed to ensure that RMS record structures are preserved as your files are moved.

Use the EXCHANGE/NETWORK command to transfer files between VMS nodes and non-VMS nodes when the differences in the file organizations would otherwise prevent the transfer or could lead to undesirable results. While COPY ensures that both the contents and the attributes of a replicated file are preserved, EXCHANGE/NETWORK is more flexible. EXCHANGE/NETWORK offers you explicit control of your record attributes during file transfers, with the opportunity to make a file usable on several different operating systems.

format

EXCHANGE/NETWORK *input-file-spec[...]* *output-file-spec*

parameters***input-file-spec[...]***

Specifies the name of an existing file to be transferred. Wildcard characters are allowed. Use a comma (,) to indicate multiple file specifications.

output-file-spec

Specifies the name of the output file into which the input is transferred.

You must specify at least one field in the output file specification. If you omit the device or directory, your current default device and directory are used. The EXCHANGE/NETWORK command replaces any other missing fields (file name, file type, version number) with the corresponding field of the input file specification.

EXCHANGE/NETWORK creates a new output file for every input file that you specify.

You can use the asterisk wildcard character in place of the following: file name, file type, or version number. The EXCHANGE/NETWORK command uses the corresponding field in the related input file to name the output file. You can also use the wildcard character in the output file specification to direct EXCHANGE/NETWORK to create more than one output file. For example:

```
$ EXCHANGE/NETWORK A.A,B.B MYPC::*C
```

This EXCHANGE/NETWORK command creates the files A.C and B.C at the non-VMS target node MYPC.

A more complete explanation of wildcard characters and version numbers follows in the “description” section.

description

The EXCHANGE/NETWORK command transfers files between VMS nodes and non-VMS nodes connected to the same DECnet network. If the non-VMS system does not support VMS file organizations, EXCHANGE/NETWORK can modify or discard file and record attributes during the transfer. However, if the target system is a VMS node, you have the option of applying new file and record attributes to the output file by supplying a File Definition Language (FDL) file, as described later in this section. EXCHANGE/NETWORK provides a number of defaults to handle the majority of transfers properly. However, in some situations, you need to know your file or record format requirements at both nodes.

VMS File and Record Attributes

All RMS files in the VMS environment include stored information, known as the file and record attributes, to describe the file and record characteristics. File attributes consist of items such as file organization, file protection, and file allocation information. Record attributes consist of items such as the record format, record size, key definitions for indexed files, and carriage control information. These attributes define the data format and access methods for the VMS RMS facility.

Non-VMS operating systems that do not support VMS file organizations have no means of storing file and record attributes with their files. Transferring a VMS file to a non-VMS system that is unable to store and handle file and record attributes can result in most of this information being discarded. Removing these attributes from a file can render it useless if it must be returned to the VMS system.

Transferring Files to VMS Nodes

When you transfer files to a VMS system from a non-VMS system, the files typically assume default file and record attributes. However, you can specify the attributes that you want the file to acquire in a File Definition Language (FDL) file. If you specify an FDL file with the /FDL qualifier, the FDL file determines the characteristics of the output file. This feature is useful in establishing compatible file and record attributes when you transfer a file from a non-VMS system to a VMS system. However, when you use an FDL file, you also assume responsibility for determining the required characteristics.

See the *VMS File Definition Language Facility Manual* for more information about FDL files.

Transferring Files to Non-VMS Nodes

EXCHANGE/NETWORK discards file and record attributes associated with a VMS file during a transfer to a non-VMS system that does not support VMS file organizations. Be aware that the loss of file and record attributes in the transfer can render the output file useless for many applications.

Selecting Transfer Modes

The EXCHANGE/NETWORK command has four transfer mode options: AUTOMATIC, BLOCK, RECORD, and CONVERT. For most file transfers, AUTOMATIC is sufficient. The AUTOMATIC transfer mode option allows EXCHANGE/NETWORK to transfer files using either block or record I/O. The selection is based on the input file organization and the operating systems involved.

Selecting the **BLOCK** transfer mode option forces **EXCHANGE/NETWORK** to open both the input and output files for block I/O access. The input file is then transferred to the output file block by block. Use this transfer mode when you transfer executable images. It is also useful when you must preserve a file's content exactly, which is a common requirement when you store files temporarily on another system or when cooperating applications exist on the systems.

Selecting the **RECORD** transfer mode option forces **EXCHANGE/NETWORK** to open both the input file and output file for record I/O access. The input file is then transferred to the output file record by record. This transfer mode is primarily used for transferring text files.

Selecting the **CONVERT** transfer mode option forces **EXCHANGE/NETWORK** to open the input file for **RECORD** access and the output file for **BLOCK** access. Records are then read in from the input file, packed into blocks, and written to the output file. This transfer mode is primarily used for transferring files with no implied carriage control. For example, to transfer a file created with **DIGITAL Standard Runoff (DSR)** to a **DECNET-DOS** system, you must use the **CONVERT** transfer mode option. To transfer the resultant output file back to a **VMS** node, use the **AUTOMATIC** transfer mode option.

Wildcard Characters

Wildcard characters are permitted in the file specifications and follow the behavior typical of other **VMS** commands with respect to the **VMS** node.

When more than one input file is specified, but wildcards are not specified in the output file specification, the first input file is copied to the output file, and each subsequent input file is transferred and given a higher version number of the same output file name. Note that the files are not concatenated into a single output file. Also note that when you transfer files to foreign systems that do not support version numbers, only one output file results, and it is the last input file.

To create multiple output files, specify multiple input files and use at least one of the following:

- An asterisk wildcard character in the output file name, file type, or version number field
- Only a node name, a device name, or a directory specification as the output file specification

When you create multiple output files, **EXCHANGE/NETWORK** uses the corresponding field from each input file in the output file name.

Use the **/LOG** qualifier when you specify multiple input and output files to verify that the files were copied as you intended.

Version Numbers

The following guidelines apply when the target node file formats accept version numbers.

If no version numbers are specified for input and output files, the EXCHANGE/NETWORK command (by default) assigns a version number to the output files that is either of the following:

- The version number of the input file
- A version number one greater than the highest version number of an existing file with the same file name and file type

When the output file version number is specified by an asterisk wildcard character, the EXCHANGE/NETWORK command uses the version numbers of the associated input files as the version numbers of the output files.

If the output file specification has an explicit version number, the EXCHANGE/NETWORK command normally uses that number for the output file specification. However, if an equal or higher version of the output file already exists, no warning message is issued, the file is copied, and the version number is set to a value one greater than the highest version number already existing.

File Protection and Creation/Revision Dates

The EXCHANGE/NETWORK command treats an output file as a new file when any portion of the output file name is explicitly specified. When the output node is a VMS system, the creation date for a new file is set to the current time and date. However, if the output file specification consists *only* of wildcard characters, the output file no longer qualifies as a new file, and, therefore, the creation date of the input file is used. That is, if the output file specification is one of the following, the creation date becomes that of the input file: *, *.* , or *.*.*.

The revision date of the output file is always set to the current time and date; the backup date is set to zero. The output file is assigned a new expiration date. (Expiration dates are set by the file system if retention is enabled; otherwise, they are set to zero.)

When the target node is a VMS node, the protection and access control list (ACL) of the output file is determined by the following parameters, in the following order:

1. Protection of previously existing versions of the output file
2. Default protection and ACL of the output directory

3. Process default file protection

For an introduction to access control lists, see the *VMS DCL Concepts Manual*.

On VMS systems, the owner of the output file usually is the same as the creator of the output file. However, if a user with extended privileges creates the output file, the owner is either the owner of the parent directory or the owner of a previous version of the output file, if one exists.

Extended privileges include any of the following:

- SYSPRV or BYPASS
- System UIC
- GRPPRV if the owner of the parent directory (or previous version of the output file) is in the same group as the creator of the new output file
- An identifier (with the resource attribute) representing the owner of the parent directory (or previous version of the output file)

qualifiers

/BACKUP

Modifies the time value specified with the */BEFORE* or */SINCE* qualifier. */BACKUP* selects files according to the dates of their most recent backup. This time qualifier is incompatible with the other time qualifiers that also allow you to select files according to time attributes: */CREATED*, */EXPIRED*, and */MODIFIED*. If you do not specify any of these four time qualifiers, the default is */CREATED*.

/BEFORE[=time]

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: *TODAY* (default), *TOMORROW*, or *YESTERDAY*. Specify one of the following time qualifiers with */BEFORE* to indicate the time attribute to be used as the basis for selection: */BACKUP*, */CREATED* (default), */EXPIRED*, or */MODIFIED*.

See the *VMS DCL Concepts Manual* for complete information about specifying time values.

/BY_OWNER[=uic]

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in the *VMS DCL Concepts Manual*.

/CONFIRM

/NOCONFIRM (default)

Controls whether a request is issued before each file transfer operation to confirm that the operation should be performed on that file. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	Ctrl/Z
1	0	ALL
	Return	

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and the Return key. QUIT or Ctrl/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplay the prompt.

/CREATED (default)

Modifies the time value specified with the /BEFORE or /SINCE qualifier. The /CREATED qualifier selects files based on their date of creation. This time qualifier is incompatible with the other time qualifiers that also allow you to select files according to time attributes: /BACKUP, /EXPIRED, and /MODIFIED. If you do not specify any of these four time qualifiers, the default is /CREATED.

/EXCLUDE=(file-spec[,...])

Excludes the specified files from the file transfer operation. You can include a directory but not a device in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

/EXPIRED

Modifies the time value specified with the /BEFORE or /SINCE qualifiers. /EXPIRED selects files according to their expiration date. (The expiration date is set with the SET FILE/EXPIRATION_DATE command.) This time qualifier is incompatible with the other time qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /MODIFIED. If you do not specify any of these four time qualifiers, the default is /CREATED.

/FDL=fdl-file-spec

Specifies that the output file characteristics are described in the File Definition Language (FDL) file. Use this qualifier when you require special output file characteristics. See the *VMS File Definition Language Facility Manual* for more information about FDL files.

Use of the /FDL qualifier implies that the transfer mode is block by block. However, the transfer mode you specify with the /TRANSFER_MODE qualifier prevails.

/LOG

/NOLOG (default)

Controls whether the EXCHANGE/NETWORK command displays the file specifications of each file copied.

When you use the /LOG qualifier, the EXCHANGE/NETWORK command displays the following for each copy operation: (1) the file specifications of the input and output files, and (2) the number of blocks or the number of records copied (depending on whether the file is copied on a block-by-block or record-by-record basis).

/MODIFIED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. The /MODIFIED qualifier selects files according to the date on which they were last modified. This time qualifier is incompatible with the other time qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /EXPIRED. If you do not specify any of these four time qualifiers, the default is /CREATED.

/SINCE[=time]

Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following time qualifiers with /SINCE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See the *VMS DCL Concepts Manual* for complete information about specifying time values.

/TRANSFER_MODE=option

Specifies the I/O method to be used in the transfer. This qualifier is useful for all file formats. You can specify any one of the following options:

Option	Function
AUTOMATIC	Allows EXCHANGE/NETWORK to determine the appropriate transfer mode.
BLOCK	Transfers block by block.
CONVERT[=option[,...]]	Reads records from the input file, packs them into blocks, and writes to the output file in block mode. The options determine what additional information is inserted during the transfer.
RECORD	Transfers record by record.

The AUTOMATIC transfer mode option allows EXCHANGE/NETWORK to determine the appropriate transfer mode. The default is the AUTOMATIC transfer mode.

If you explicitly select the BLOCK transfer mode option, EXCHANGE/NETWORK opens both the input and output files for block I/O. EXCHANGE/NETWORK then transfers the files block by block.

If you explicitly select the RECORD transfer mode option, EXCHANGE/NETWORK opens both the input and output files for record I/O. The target system must support record operations, and the input file must be record oriented.

If you select the CONVERT transfer mode option, EXCHANGE/NETWORK reads records in from the input file, packs them into blocks, and writes them to the output file in block mode. There are four options available with the CONVERT transfer mode to control the insertion of special characters in the records, as explained in the following paragraphs:

- CARRIAGE_CONTROL
- COUNTED
- FIXED_CONTROL
- RECORD_SEPARATOR=separator

If you specify CARRIAGE_CONTROL, any carriage control information in the input file is interpreted, expanded into actual characters, and included with each record.

If you specify COUNTED, the length of each record in bytes is included at the beginning of each record. The length includes all FIXED_CONTROL, CARRIAGE_CONTROL, and RECORD_SEPARATOR information in each record.

If you specify FIXED_CONTROL, all variable length with fixed control record (VFC) information is written to the output file as part of the data. This information follows the record length information if the COUNTED option was specified.

If you specify `RECORD_SEPARATOR`, a 1- or 2-byte record separator is inserted between each record. Record separator characters are the last characters in the record. The three choices for separator characters are CR for carriage return only, LF for line feed only, or CRLF for carriage return and line feed.

example

```
$ EXCHANGE/NETWORK VMS_FILE.DAT FOO::FOREIGN_SYS.DAT
```

In this example, the `EXCHANGE/NETWORK` command transfers the file `VMS_FILE.DAT` located in the current default device and directory to the file `FOREIGN_SYS.DAT` on the non-VMS node `FOO`. Because the `/TRANSFER_MODE` qualifier was not explicitly specified, `EXCHANGE/NETWORK` automatically determines whether the transfer method should be block or record I/O.

```
$ EXCHANGE/NETWORK/TRANSFER_MODE=BLOCK -
_$ FOO::FOREIGN_SYS.DAT VMS_FILE.DAT
```

In this example, the `EXCHANGE/NETWORK` command transfers the file `FOREIGN_SYS.DAT` from the non-VMS node `FOO` to the file `VMS_FILE.DAT` in the current default device and directory. Block I/O is specified for the transfer mode.

```
$ EXCHANGE/NETWORK/FDL=VMS_FILE_DEFINITION.FDL -
_$ FOO::REMOTE_FILE.TXT VMS_FILE.DAT
```

In this example, the `EXCHANGE/NETWORK` command transfers the file `REMOTE_FILE.TXT` on node `FOO` to the file `VMS_FILE.DAT`. The file attributes for the output file `VMS_FILE.DAT` are obtained from the File Definition Language (FDL) source file `VMS_FILE_DEFINITION.FDL`. For more information about creating FDL files, see the *VMS File Definition Language Facility Manual*. Because the qualifier `/FDL` is specified and the `/TRANSFER_MODE` qualifier is omitted, the transfer mode uses block I/O, by default.

```
$ EXCHANGE/NETWORK -
_$ /TRANSFER_MODE=CONVERT=(CARRIAGE_CONTROL, COUNTED, -
_$ RECORD_SEPARATOR=CRLF, FIXED_CONTROL) -
_$ PRINT_FILE.TXT FOO::*
```

In this example, the `EXCHANGE/NETWORK` command transfers the file `PRINT_FILE.TXT` from the current default device and directory to the file `PRINT_FILE.TXT` on the non-VMS node `FOO`. The use of the `CONVERT` option with the `/TRANSFER_MODE` qualifier forces the input file to be read in record by record, modified as specified by the convert options described below, and written to the output file block by block. As many records as will fit are packed into the output blocks.

The CONVERT option CARRIAGE_CONTROL specifies that carriage control information be converted to ASCII characters and inserted before the data or appended to the record, depending on whether prefix control or postfix control, or both, are used. The CONVERT option FIXED_CONTROL specifies that any fixed control information be translated to ASCII characters and inserted at the beginning of the record. The CONVERT option RECORD_SEPARATOR=CRLF appends the two specified characters, carriage return and line feed, to the end of the record. The CONVERT option COUNTED specifies that the total length of the record must be counted (once the impact of all the previous convert options have been added), and the result is to be inserted at the beginning of the record, in the first two bytes.

Index

A

- ALF (auto-login facility)
 - adding a new record, 2-6
 - displaying a record, 2-8
 - function of, 2-6
 - removing a record, 2-7
- ALF ADD command, 2-6
- ALF REMOVE command, 2-7
- ALF SHOW command, 2-8
- Alternate working device option
 - See AWD option
- ANALYZE/AUDIT command, 1-2
- Applications
 - running locally, 1-5
 - running remotely, 1-5
- ASK callback, 3-10
- AST
 - in target process, 4-26
- Audit server, 2-3
- Auto-login facility
 - See ALF
- AWD option, 3-11

B

- /BACKUP qualifier
 - EXCHANGE/NETWORK command, A-23
- Backup Utility, 2-15 to 2-19
 - /BUFFER_COUNT command qualifier, 2-18
 - cyclic redundancy checking, 2-18
 - performance enhancements, 2-15
 - pressing Ctrl/T during BACKUP, 2-19

Backup Utility (cont'd.)

- setting SYSGEN parameters to enhance performance of, 2-17
- setting up BACKUP account, 2-15
- summary of new features, 2-15
- UAF parameters for BACKUP account, 2-16
- /BEFORE qualifier
 - EXCHANGE/NETWORK command, A-23
- Buffer
 - \$GETJPI
 - using for multiple requests for information, 4-40
- BYTLM quota
 - using with \$GETJPI buffers, 4-40
- /BY_OWNER qualifier
 - EXCHANGE/NETWORK command, A-23

C

- Callbacks
 - ASK, 3-10
 - CHECK_NETWORK, 3-3
 - CHECK_NET_UTILIZATION, 3-8
 - CHECK_PRODUCT_VERSION, 3-4
 - COMPARE_IMAGE, 3-5
 - CREATE_DIRECTORY, 3-9
 - GET_IMAGE_ID, 3-6
 - UPDATE_IDENTIFIER, 3-7
- CHECK_NETWORK callback, 3-3
- CHECK_NET_UTILIZATION callback, 3-8
- CHECK_PRODUCT_VERSION callback, 3-4
- COMPARE_IMAGE callback, 3-5

Index-2

Compound document
 See also DDIF
 defined, A-1
CONFIGURATION SHOW CLUSTER
 command, 2-9
/CONFIRM qualifier
 EXCHANGE/NETWORK command,
 A-24
/CREATED qualifier
 EXCHANGE/NETWORK command,
 A-24
CREATE_DIRECTORY callback, 3-9
Cyclic redundancy checking, 2-18

D

DCL commands
 ANALYZE/AUDIT, 1-2
 DISMOUNT, 1-11
 SET AUDIT, 1-11
 SET DISPLAY, 1-3
 SET FILE, 1-20
 SHOW AUDIT, 1-21
 SHOW DISPLAY, 1-8
 SHOW SYSTEM, 1-23
 SHOW USERS, 1-23
DDIF (Digital Document Interchange
 Format)
 VMS RMS support of, A-1
DDIF-to-text RMS extension, A-1
Debugger, 3-1
DECnet
 running DECwindows applications
 across, 1-5
DECnet account
 limiting default access, 2-12
DECW\$DISPLAY, 1-3
DEINSTALL command, 2-5
Device
 creating, 1-3
 modifying, 1-3
 scanning of across the cluster, 4-32
Digital Document Interchange Format
 See DDIF
DISMOUNT command, 1-11

E

ERLBUFFERPAGES parameter
 description, 2-6
EXCHANGE/NETWORK command,
 A-18 to A-28
 creating files, A-22
 protecting files, A-22
 qualifiers, A-23
 selecting transfer modes, A-20
 transferring files, A-20
 wildcard characters, A-21
/EXCLUDE qualifier
 EXCHANGE/NETWORK command,
 A-24
/EXPIRED qualifier
 EXCHANGE/NETWORK command,
 A-24

F

F\$CONTEXT lexical function, 1-24 to
 1-29
F\$FILE_ATTRIBUTES item code
 AI, 1-29
 BI, 1-29
 ERASE, 1-29
 FFB, 1-29
 GBC, 1-29
 JOURNAL_FILE, 1-29
 LOCKED, 1-29
 RU, 1-29
 STORED_SEMANTICS, 1-29
F\$PID lexical function, 1-24
FAL (file access listener)
 default access, 2-12
/FDL qualifier, A-25
File
 copying, A-18
 creating, A-18
 transferring, A-18, A-20
File access listener
 See FAL

File protection

with EXCHANGE/NETWORK
command, A-22

File tag

creating, A-1
DDIF, A-1
disposition by COPY command, A-4
requirement for, A-1
stored semantics file attribute, A-1
using, A-1

G

GET_IMAGE_ID callback, 3-6

K

Kit debug option

See K option

K option, 3-12

L

LICENSE LOAD command, 2-9

License Management Facility

See LMF

Licenses

manipulating with SYSMAN, 2-9

LICENSE UNLOAD Command, 2-10

LMF (License Management Facility)
changes, 2-11

Logical names

use in SYSMAN, 2-10

LOGIN procedure

/NEW_PASSWORD qualifier, 1-11

/LOG qualifier

EXCHANGE/NETWORK command,
A-25

Loopback mirror

See MIRROR

M

MAIL

default access, 2-12

MIRROR

default access for loopback testing,
2-12

Modes

of transferring files, A-20

/MODIFIED qualifier

EXCHANGE/NETWORK command,
A-25

Monitor Utility

cluster performance, 2-13

Multicast address

finding with SYSMAN, 2-9

N

NETCONFIG.COM command procedure

security enhancements, 2-11

NETCONFIG_UPDATE.COM, 2-14

Network default access

controlling access to your system, 2-11

for existing systems, 2-14

for VAXcluster members, 2-14

/NEW_PASSWORD qualifier

LOGIN procedure, 1-11

/NOCONFIRM qualifier

EXCHANGE/NETWORK command,
A-24

/NOLOG qualifier

EXCHANGE/NETWORK command,
A-25

N option, 3-12

P

Page file

deinstalling, 2-5

PHONE

default access, 2-12

PID (process identification) number

defined, 4-6, 4-7

Index-4

PID (process identification) number (cont'd.)

using to reference remote process, 4-6,
4-7

Process

See also Remote process

See also SYS\$GETJPI system service

See also SYS\$PROCESS_SCAN system
service

locating a subset of, 4-36

obtaining information about, 4-6
example, 4-10
synchronously, 4-22

obtaining information about one
process, 4-9, 4-10

obtaining information about processes
on specific nodes, 4-20, 4-21

obtaining information about the calling
process, 4-10

obtaining information about using PID,
4-6

obtaining information about using
process name, 4-6, 4-7, 4-8

scanning across the clusters, 4-36
using \$PROCESS_SCAN item list to
specify selection criteria about,
4-14, 4-15
example, 4-18, 4-19

using \$PROCESS_SCAN item list with
remote procedures, 4-23

using \$PROCESS_SCAN search for,
4-14

using wildcard search for, 4-12

Process context

using with \$GETJPI, 4-6

Process name

length of for remote processes, 4-7,
4-8

specifying for local process, 4-7, 4-8
specifying for remote processes, 4-7,
4-8

specifying processes by, 4-47

specifying processes with node name,
4-45

Process name (cont'd.)

using to obtain information about
remote processes, 4-6, 4-7, 4-8,
4-19

example, 4-11

Process search, 4-36

obtaining information about one
process, 4-9, 4-10

obtaining information about the calling
process, 4-10

searching on all nodes, 4-21

searching on specific nodes, 4-20,
4-21

using \$PROCESS_SCAN item list to
specify selection criteria about
processes, 4-14, 4-15

example, 4-18, 4-19

using item list with remote procedures,
4-23

using item-specific flags to control
selection information, 4-14

using wildcard on local system, 4-12

\$PROCESS_SCAN system service

controlling selection information for
\$GETJPI, 4-38

item descriptor

buffer length, 4-37

format, 4-37

using item-specific flags, 4-38

R

Release notes option

See N option

Remote process

See Process

Remote process search

See Process search

Restore save set and pause option

See RSP option

RIGHTSLIST database, 2-8

RSP option, 3-11

S

Security

- audit server, 2-3

Security auditing

- new features, 2-1

Security enhancements

NETCONFIG.COM

- for existing systems, 2-14

- for new systems, 2-11

SET ASK_CASE option, 3-7

SET AUDIT command, 1-11

SET DISPLAY command, 1-3

SET ENVIRONMENT/NODE command,
2-10

SET FILE command

- /[NO]SEMANTICS qualifier, 1-20

SET POSTINSTALL option, 3-8

SHOW AUDIT command, 1-21

SHOW DISPLAY command, 1-8

SHOW SYSTEM command

- /CLUSTER qualifier, 1-23

- /NODE[=(name...)] qualifier, 1-23

SHOW USERS command

- /BATCH qualifier, 1-23

- /CLUSTER qualifier, 1-23

- /FULL qualifier, 1-23

- /INTERACTIVE qualifier, 1-23

- /NETWORK qualifier, 1-23

- /NODE[=name(...)] qualifier, 1-23

- /SUBPROCESS qualifier, 1-23

/SINCE qualifier

- EXCHANGE/NETWORK command,
A-25

Stored semantics file attribute

- See File tag

Swap file

- deinstalling, 2-5

SYS\$CANWAK system service, 4-5

SYS\$DELPRC system service, 4-5

SYS\$DEVICE_SCAN system service,
4-1, 4-31, 4-32

SYS\$DISMOUNT system service, 4-4

SYS\$FORCEX system service, 4-5

SYS\$GETJPI system service, 4-5, 4-6

SYS\$GETJPI system service (cont'd.)

- See also SYS\$PROCESS_SCAN system
service

- AST in target process, 4-26

- buffer, 4-23, 4-25

- control flags, 4-26

- defined, 4-6

- item codes, 4-9

- item list, 4-14, 4-23

- item-specific flags, 4-14

- obtaining information about all
processes on the local system,
4-9, 4-12

- obtaining information about one
process, 4-9

- obtaining information with
\$PROCESS_SCAN context, 4-9

- obtaining information with PID, 4-9

- obtaining information with wildcard
search
example, 4-12

- packing information in buffers, 4-23,
4-25

- searching for processes on all nodes,
4-21

- searching for processes on specific
nodes, 4-20, 4-21

- searching for selected processes, 4-14

- specifying buffer size, 4-23, 4-25

- specifying criteria to select processes
example, 4-19

- swapping processes, 4-26

- synchronizing calls, 4-20, 4-21, 4-22

- using \$PROCESS_SCAN item list to
specify selection criteria about
processes, 4-14, 4-15, 4-18, 4-19

- using \$PROCESS_SCAN item-
specific flags to control selection
information, 4-14

- using \$PROCESS_SCAN search, 4-14

- using item list to specify selection
criteria about processes
example, 4-18

- using item list with remote procedures,
4-23

Index-6

SYS\$GETJPI system service (cont'd.)

- using multiple \$PROCESS_SCAN contexts, 4-22
- using synchronous calls, 4-22
- using wildcard
 - example, 4-12
- using wildcard as pidadr, 4-9, 4-12
- using wildcard search, 4-12

SYS\$GETUAI system service, 4-1

- authorization flags, 4-3
- new item codes, 4-3

SYS\$MOUNT system service, 4-4

SYS\$PROCESS_SCAN system service, 4-1, 4-6, 4-31, 4-36

- See also SYS\$GETJPI system service defined, 4-6
 - obtaining information about processes on all nodes, 4-21
 - obtaining information about processes on specific nodes, 4-20, 4-21
 - searching on all nodes, 4-21
 - searching on specific nodes, 4-20, 4-21
 - setting up multiple contexts, 4-22
 - specifying selection criteria about processes
 - example, 4-18
 - using \$PROCESS_SCAN item list to specify one selection criterion about processes
 - example, 4-15
 - using item list to control selection information
 - example, 4-18
 - using item list to specify selection criteria about processes, 4-14, 4-15
 - example, 4-18, 4-19
 - using item list with remote procedures, 4-23
 - using item-specific flags to control selection information, 4-14
 - example, 4-15
- ### SYS\$RESUME system service, 4-5
- ### SYS\$SCHDWK system service, 4-5
- ### SYS\$SETPRI system service, 4-5

SYS\$SETUAI system service, 4-1

- authorization flags, 4-3
- new item codes, 4-2

SYS\$SUSPEND system service, 4-5

SYS\$WAKE system service, 4-5

SYSGEN Utility, 2-5 to 2-6

- DEINSTALL command, 2-5
- ERLBUFFERPAGES parameter, 2-6

SYSTEMAN Utility

- auto-login facility, 2-6
- cluster privileges, 2-8
- new and changed features, 2-6
- using to manipulate licenses, 2-9

System Generation Utility

- See SYSGEN Utility

System object

- default access for, 2-12

System parameter

- description, 2-6

System service, 4-1, 4-2, 4-3, 4-5

- obtaining information about processes, 4-6

System user authorization file

- See SYSUAF

SYSUAF (system user authorization file), 2-8

T

TASK object

- restricting default access, 2-12

Transfer modes

- EXCHANGE/NETWORK command, A-20

/TRANSFER_MODE qualifier

- EXCHANGE/NETWORK command, A-25

U

UETP (User Environment Test Package)

- testing the DECnet connection, 2-12

UPDATE_IDENTIFIER callback, 3-7

User Environment Test Package

- See UETP

User-written programs and procedures
default access for, 2-12

V

Version number
assigning, A-22
VMSINSTAL, 3-3 to 3-12
command line changes, 3-10
VMS Performance Monitor
See VPM
VPM (VMS Performance Monitor), 2-13
default access for, 2-12

W

Wildcards
EXCHANGE/NETWORK command,
A-21
Wildcard search
obtaining information about processes,
4-36
example, 4-12
using \$GETJPI, 4-12

How to Order Additional Documentation

Technical Support

If you need help deciding which documentation best meets your needs, call 800-343-4040 before placing your electronic, telephone, or direct mail order.

Electronic Orders

To place an order at the Electronic Store, dial 800-DEC-DEMO (800-332-3366) using a 1200- or 2400-baud modem. If you need assistance using the Electronic Store, call 800-DIGITAL (800-344-4825).

Telephone and Direct Mail Orders

Your Location	Call	Contact
Continental USA, Alaska, or Hawaii	800-DIGITAL	Digital Equipment Corporation P.O. Box CS2008 Nashua, New Hampshire 03061
Puerto Rico	809-754-7575	Local DIGITAL subsidiary
Canada	800-267-6215	Digital Equipment of Canada Attn: DECdirect Operations KAO2/2 P.O. Box 13000 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6
International	—————	Local DIGITAL subsidiary or approved distributor
Internal ¹	—————	SDC Order Processing - WMO/E15 <i>or</i> Software Distribution Center Digital Equipment Corporation Westminster, Massachusetts 01473

¹For internal orders, you must submit an Internal Software Order Form (EN-01740-07).

Reader's Comments

VMS Version 5.2 New Features Manual
AA-LA97B-TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

I rate this manual's:

	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Completeness (enough information)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Clarity (easy to understand)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Organization (structure of subject matter)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Figures (useful)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Examples (useful)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Index (ability to find topic)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Page layout (easy to find information)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

I would like to see more/less _____

What I like best about this manual is _____

What I like least about this manual is _____

I found the following errors in this manual:

Page	Description
_____	_____
_____	_____
_____	_____

Additional comments or suggestions to improve this manual:

I am using **Version** _____ of the software this manual describes.
Name/Title _____ Dept. _____

Company _____ Date _____

Mailing Address _____

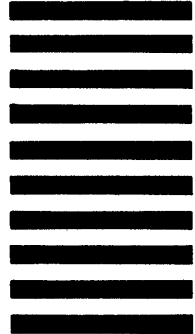
Phone _____

- Do Not Tear - Fold Here and Tape

digital™



No Postage
Necessary
if Mailed
in the
United States



BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
Corporate User Publications—Spit Brook
ZK01-3/J35
110 SPIT BROOK ROAD
NASHUA, NH 03062-9987



- Do Not Tear - Fold Here

Cut Along Dotted Line

Reader's Comments

VMS Version 5.2 New Features Manual
AA-LA97B-TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Completeness (enough information)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Clarity (easy to understand)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Organization (structure of subject matter)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Figures (useful)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Examples (useful)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Index (ability to find topic)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Page layout (easy to find information)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

I would like to see more/less _____

What I like best about this manual is _____

What I like least about this manual is _____

I found the following errors in this manual:

Page	Description
_____	_____
_____	_____
_____	_____

Additional comments or suggestions to improve this manual:

I am using **Version** _____ of the software this manual describes.
Name/Title _____ Dept. _____

Company _____ Date _____

Mailing Address _____
_____ Phone _____

Do Not Tear - Fold Here and Tape

digital™

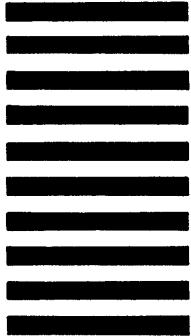


No Postage
Necessary
if Mailed
in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
Corporate User Publications—Spit Brook
ZK01-3/J35
110 SPIT BROOK ROAD
NASHUA, NH 03062-9987



Do Not Tear - Fold Here

Cut Along Dotted Line