VMS SES
User's Guide
Version 5.2

Order Number: QS-970AA-UG

November 1989

This manual introduces Version 5.2 of the VMS Security Enhancement Service to the general user. In addition, this manual describes the command language, system routines, error messages, and terms of the Version 5.2 VMS Security Enhancement Service software.

Revision/Update Information:	This manual supersedes the Version 5.1 *VMS SES User's Guide*.
Operating System and Version:	VMS Version 5.2
Software Version:	SEVMS Version 5.2

Digital Equipment Corporation

November 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.

Restricted Rights: Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

The postpaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DEC
DEC/CMS
DEC/MMS
DECnet
DECsystem–10
DECSYSTEM–20
DECUS
DECwriter

DIBOL
EduSystem
IAS
MASSBUS
PDP
PDT
RSTS
RSX

UNIBUS
VAX
VAXcluster
VMS
VT
digital ™

This document was prepared using VAX DOCUMENT, Version 1.1

Contents

FIGURES

TABLES

Preface

This document introduces the VMS Security Enhancement Service and describes the command language interface, system routines, error messages, and terms associated with it.

The VMS Security Enhancement Service (VMS SES) is a software security consulting package. It provides many features of mandatory access controls and security auditing for the VMS operating system.

The VMS SES software security consulting package is composed of the following components:

- Services performed by a DIGITAL consultant
- Licensed software
- Documentation

VMS SES provides the services of a trained DIGITAL consultant who supports the customer in several areas, such as: assisting in planning security policies and controls, training users, and installing the licensed software.

The licensed software component of this product is called SEVMS. SEVMS provides a tool set for devising a system-wide security policy to help safeguard users, data, and software from security threats. Since this manual describes the features of the licensed software, the term SEVMS is used throughout this manual to reference this software. SEVMS is also the VMS facility name for the licensed software and is used as a prefix for many of the software components.

A documentation set which describes the SEVMS software and how it is installed, used, and managed is provided with the VMS SES package.

Intended Audience

This manual is intended for all users of an SEVMS system. It is assumed that users of this manual have a general working knowledge of VMS.

Document Structure

The information in this manual is divided into the following chapters:

- Introduction to VMS Security Enhancement Service — This chapter contains general information about SEVMS, information about complementary security techniques, mandatory access controls, access rules, logging into a system, classifying directories and files, and propagation of classification.
- DCL Commands — This chapter describes the DCL commands which are used by SEVMS and explains how these commands operate.

- Programming Information — This chapter provides programming information, including an explanation of the SEVMS class block format and binary audit record format, descriptions of the system service routines that are employed by SEVMS, and a description of SEVMS manadatory access control items which are used with VMS System Services.
- System Messages — This section contains system messages which are unique to SEVMS.
- Glossary — This section explains the terms most commonly used in SES documentation.

Associated Documents

This manual should be used in conjunction with the other manuals of the SES document set and the manuals of the VMS document set. References will be made throughout this manual to VMS SES manuals and VMS manuals.

SES Document Set

This manual is one of three manuals that form the VMS Security Enhancement Service (SES) document set. This document set consists of the following manuals:

- *VMS SES User's Guide* — This manual describes the mandatory protection mechanisms provided by SEVMS, the interaction of these mechanisms with VMS discretionary protection mechanisms, and the use of commands and utilities which are unique to SEVMS. It is intended for all SEVMS users.
- *VMS SES Security Manager's Guide* — This manual describes the configuration, management, and operation of SEVMS. It is intended for use by system administrators and security officers. This manual assumes that the reader is familiar with basic VMS security practices and the VMS documentation which describes VMS security.
- *VMS SES Installation Guide and Release Notes* — This manual is intended as a supplemental manual of the SES documentation set. It provides information concerning the installation (but not configuration) of SEVMS on a VMS system. It also contains release notes which summarize omitted features, resolved problems, new features, and known problems and restrictions for the current release of the SEVMS software.

Together, these manuals form complete documentation about the SEVMS software. For information about related VMS features and functions, the user should refer to the manuals of the VMS document set.

VMS Documentation Set

The VMS documentation set has two main divisions:

- VMS Base Documentation Set
- VMS Extended Documentation Set

The VMS Base Documentation Set is a desk-top set for users of small standalone systems and low-end Local Area VAXclusters, and for general users of large VAX systems. The Base Documentation Set contains concise, easy to find, information about performing day-to-day tasks. This documentation set contains the following components:

- Overview of VMS Documentation
- VMS New Features Manual
- VMS General User's Manual
- VMS System Manager's Manual
- VMS Mini-Reference Manual
- VMS License Management Manual

The VMS Extended Documentation Set is a full documentation set for users who need more detail about any VMS component to perform daily tasks. The Extended Documentation Set also meets the needs of system managers of large VAX systems and of system and application programmers.

This documentation set contains the following components:

- General User Subkit
- System Management Subkit
- Programming Subkit

These manuals are supplemented by several other forms of VMS documentation: Release Notes, Obsolete Features Kit, Software Installation and Operations Guides, online help information, and other optional documentation.

Refer to the *Overview of VMS Documentation* booklet in the VMS documentation set for complete information about the VMS documentation set.

Relationship Between VMS and SEVMS Documentation

The documentation for SEVMS is intended to be used along with the documentation for VMS. While the SES documentation set addresses issues specific to the SEVMS product, issues of a more general nature pertaining to VMS are addressed in the VMS documentation set. Therefore, you can consider the manuals of the SES document set to be an extension of your existing VMS document set. As such, SES manuals do not repeat information already contained in existing VMS documentation. Instead, references are made throughout SES manuals to several of the manuals in the VMS document set, when appropriate. The following VMS documentation is most frequently referenced by the SES manuals:

- *VMS System Management Subkit*
- *Guide to VMS System Security*
- *VMS Audit Analysis Utility Manual*
- *VMS DCL Dictionary*
- *VMS Release Notes*

Conventions

This section describes the VMS and SEVMS conventions which are used in this manual.

VMS Conventions Used in This Manual

Throughout this manual, the following standard conventions are used in examples of commands:

Convention	Meaning
[ ]	Square brackets indicate that the enclosed item is optional.
{ }	Braces enclose a list from which one element must be chosen.
< >	Angle brackets indicate that item is to be replaced by a specific instance of the named quantity.
|	The OR symbol separates alternatives within braces or brackets.
. . .	An ellipsis indicates that the preceding item(s) can be repeated one or more times.
:=	A "colon equals" indicates the item to its left is defined as the item to its right.

Unless otherwise indicated in the examples, commands are terminated by pressing the Return key.

Colons (:) and equals signs (=) are used interchangeably in descriptions of DCL command qualifiers.

SEVMS Conventions Used in This Manual

SEVMS mandatory access controls introduce a number of new protection attributes and relationships. Among these are the concepts of hierarchical *levels* and non-hierarchical *categories*. Categories form discrete mathematical *sets*.

The *operators* used to indicate the relationship between numeric quantities *(scalar)* differ from the operators used to indicate the relationship between non-numeric quantities *(sets)*, although their meanings are similar. The operators used in this manual are described and compared in the following table.

Operator	Scalar interpretation for Security Levels	Operator	Set interpretation f Security Categories
$<$	is less than	$\subset$	is a proper subset of
$\leq$	is less than or equal	$\subseteq$	is a subset of
$=$	is equal to	$\equiv$	is identical to
$>$	is greater than	$\supset$	is a proper superset of
$\geq$	is greater than or equal	$\supseteq$	is a superset of
$\neq$	is not equal to	$\not\equiv$	is not identical to

In informal discussions of the relationship between two classifications, the scalar relationships may be used to refer to both the scalar (level) and set (categories) portions of the classification. For instance, the informal statement "A's classification is equal to B's" means "A's level $=$ B's level AND A's categories $\equiv$ B's categories".

Dominates describes a relationship between two classifications. "A's classification dominates B's" means "A's level $\geq$ B's level **AND** A's categories $\supseteq$ B's categories".

1 Introduction to SEVMS

This chapter introduces and defines the software of the VMS Security Enhancement Service (VMS SES), which is referred to in this manual as SEVMS.

1.1 Overview

This section briefly describes the nature of SEVMS, its intent, and its advantages.

1.1.1 Description of SEVMS

SEVMS implements a *mandatory* (i.e. *non-discretionary*) access control mechanism. The most distinguishing characteristic of the SEVMS mandatory access control mechanism is that it is an implementation of a security policy which is *beyond direct user control.* This security policy is centrally and uniformly established by the system security manager (often the system manager). SEVMS is responsible for enforcing the security policy established by the security manager.

1.1.2 Intent of SEVMS

It is important to note that the SEVMS mandatory controls do not replace the familiar VMS discretionary access controls (such as Access Control Lists). Instead, SEVMS mandatory controls are used *in addition to* standard VMS discretionary controls, and *augment* VMS protection mechanisms. Therefore, SEVMS is intended to provide the system security manager with a means to enforce an additional system-wide mandatory security policy.

1.1.3 Advantages of SEVMS

The use of SEVMS mandatory access controls enables the security manager to classify users and data with different levels of sensitivity. By classifying users, SEVMS ensures the following:

- Users cannot read data unless their classification permits it.
- Users cannot write data with a new classification that would grant read access to users who could not previously read the data.

In addition to the advantages of classifying users, SEVMS also provides the following advantages:

- It provides auditing of attempts to compromise mandatory access controls.
- It restricts certain users to certain terminals, based on classification.

- It restricts printed output to certain printers, based on classification.
- It restricts creation of files on certain disks, based on classification.
- It provides a uniform mechanism for the sensitivity labeling of print jobs.

1.2 Complementary Security Techniques of VMS and SEVMS

As mentioned at the beginning of this chapter, SEVMS security features are *not* meant to replace standard VMS security features. Instead, the security features of SEVMS are provided in addition to those standard features of VMS. Before implementing SEVMS security features, it is important that the security manager be very familiar with VMS security features and how they are used. Therefore, the material contained in the *Guide to VMS System Security* should be thoroughly read and understood.

1.3 Mandatory Access Controls

Mandatory access controls in SEVMS are used to control access between *subjects* and *objects* in a system. An example of a subject is a process. (In fact, under SEVMS, processes are the only subjects.) An example of an object is a file. Subjects and objects can be assigned *classification labels* which are made up of a combination of hierarchical *secrecy levels* and non-hierarchical *secrecy categories*. Classification labels are discussed in more detail in Section 1.3.4 of this manual.

The SEVMS software provides the following mandatory access control elements:

- 256 secrecy levels
- 128 secrecy categories
- 256 integrity levels
- 64 integrity categories

Subjects, objects, and classification labels are further discussed in "Chapter 2, Overview of SEVMS" of the *VMS SES Security Manager's Guide*. Mandatory access controls, secrecy levels, and secrecy categories are discussed in greater detail in "Chapter 4 - Mandatory Access Control and the User" and "Chapter 5 - Mandatory Access Control and the Security Manager" of the *VMS SES Security Manager's Guide*.

The remainder of this section discusses the following topics:

- Secrecy levels and categories
- Integrity levels and categories
- Classification labels

1.3.1 Secrecy Levels

SEVMS secrecy levels are hierarchical - they range in value from 0 to 255, with 0 representing the lowest level and 255 representing the highest. A lower secrecy level indicates a lower classification; a higher secrecy level represents a higher classification. Secrecy levels can always be specified as numeric values; however, secrecy levels are normally represented symbolically. For example, the secrecy level 0 might be symbolically represented by UNCLASSIFIED, the secrecy level 10 might be represented by CONFIDENTIAL, and a secrecy level of 255 might be TOP SECRET, and so on. The assignments are made by the system manager or security manager.

1.3.2 Secrecy Categories

Secrecy categories are non-hierarchical. As with secrecy levels, secrecy categories can be represented numerically (1 through 128), or they can be represented symbolically. Secrecy categories are used to assign non-hierarchical attributes to an object. Secrecy categories are especially useful when there is a need to create separate compartments for information. Because secrecy categories are disjoint, no category can be considered more classified than another; only the presence or absence of a particular secrecy category matters.

For instance, a secrecy category RED document is more classified than a secrecy category BLUE document because the BLUE document does not have the RED secrecy category. But, the reverse is also true; the same can be said of the BLUE document with respect to the RED document.

To avoid the confusion of the situation described above, the term *dominate* should be used when comparing two classifications. For example, if you are comparing classifications "A" and "B", you would use the following terminology: The statement *"A's" classification dominates B's"* means that "A's" secrecy level is greater than "B's" and "B's" set of secrecy categories is a subset of "A's". Although "dominate" is the more accurate term, comparative terms such as "greater than" are often used informally to mean the same thing.

1.3.3 Integrity Levels and Categories

There are 256 *integrity levels* and 64 *integrity categories*. Although SEVMS provides controls for integrity, the consequences of its use with VMS are not well understood. Therefore, DIGITAL supports the detailed functionality of SEVMS in regards to integrity controls, but cannot make any assurances as to the overall consequences of using those controls. In particular, **the VMS Security Enhancement Service does not provide assistance in the use of integrity at this time.** Integrity has been omitted from this discussion of mandatory access controls for the sake of simplicity. See the *VMS SES Security Manager's Guide* for more information on integrity.

1.3.4 Classification Labels

A *classification label* (also referred to as "label" in this manual) consists of a secrecy level and some combination of secrecy categories. Labels can be assigned to the following objects and subjects:

- ODS-2 files
- ODS-2 disk volumes
- Mailboxes
- Shared logical name tables
- Global sections
- Disk devices
- Logical name tables
- Magnetic tape devices
- Terminal devices
- Printer devices
- Processes
- Queues

Note: **Any device can be labeled - SEVMS will restrict access to that device at the QIO level; however, SEVMS provides specific multilevel device support for only the devices listed above.**

The following example illustrates the assignment of a label.

User Jones may log in at the level SECRET with categories RED, WHITE, and BLUE as follows:

```
USERNAME: JONES/SECRECY=(LEVEL:SECRET,CATEGORY:(RED,WHITE,BLUE))
PASSWORD:

                Welcome to VMS V5.2...
.
.
.
```

In this case, the specified classification label is attached to the process JONES for the lifetime of the process.

1.4 Mandatory Access Control Rules

This section describes the rules of mandatory access control and the privileges which affect these rules.

1.4.1 Access Rules Description

Mandatory access control rules are straightforward and simple. Put informally, an unprivileged user is allowed to do the following:

- READ DOWN (read a lower classification)
- WRITE UP (write to a higher classification)
- READ/WRITE EQUAL (read and write to an equal classification)

These rules are summarized in more formal terms in Table 1-1.

Table 1-1 Mandatory Access Control Rules

If your process classification	Permitted access is
DOMINATES the object classification	READ ONLY
is DOMINATED BY the object classification	WRITE ONLY†
is EQUAL TO object classification	READ and WRITE

†Some objects, particularly files, cannot be accessed WRITE ONLY.

1.4.2 Access Rules and Privileges

The mandatory access control rules can be circumvented by users with certain privileges.

SEVMS does not remove or limit existing VMS privileges. Users with any of the ALL privileges (as defined in the *Guide to VMS System Security*), can manage to BYPASS the mandatory access controls, just as they can bypass the discretionary access controls.

The privileges which directly affect mandatory access control rules are as follows:

- DOWNGRADE - Allows a process to write to a lower secrecy object or to lower an object's classification.
- SECURITY - Allows a process to set a multilevel classification and to turn on and off security auditing.
- BYPASS - Bypasses all protection checks.
- VOLPRO - Bypasses volume protection checks.
- UPGRADE - Allows a process to write to a higher integrity object or to raise the integrity of an object.
- READALL - Allows read access to everything.

To be assured that access rules are enforced, it is important that the security manager be familiar with VMS privileges and restrict their use.

1.5 Relationship Between Discretionary and Mandatory Access Controls

In the SEVMS implementation of mandatory access controls (MAC), there are two types of access: *read*, and *write*. In the VMS (and thus SEVMS) implementation of discretionary access controls (DAC), there are at least 7 access types: READ, WRITE, EXECUTE, DELETE, CONTROL, PHYSICAL, and LOGICAL.

There are no fixed relationships between the MAC and DAC access types; whether a DAC access type requires a particular MAC access type depends upon the particular object and operation.† This can be confusing, unless you understand the interpretations of the DAC and MAC access types.

The DAC access types are described in Section 4.2.7 of the *Guide to VMS System Security*. The MAC access types are described below, in Section 1.5.1.

Reminder: **Both MAC and DAC checks must be passed before access to an object is allowed.**

1.5.1 Mandatory Access Types

There are two MAC (mandatory access controls) access types, *read* and *write*.

These MAC access types have the following interpretations for all objects:

READ
The right to observe the contents and attributes of an object.

WRITE
The right to modify the contents and attributes of an object.

"MAC access types" refer to the *kind* of operation being performed on an object, not to the DAC access types. For instance, to run a program, you need MAC READ access to the image as well as DAC EXECUTE access. In this case, running the program reads the executable image into the processes' address space, allowing the user to see it - so MAC READ access is required.

Note: **VMS (thus SEVMS) does not support write-only access to some objects, particularly files. If write-only access is not supported for an object, then both READ and WRITE MAC access is required to write to the object.**

MAC access types, unlike the DAC access types, are not explicitly associated with object protection codes or ACLs. For instance, you can't grant a user mandatory READ access to an object by way of an ACL. It is the relationship between the subject's and the object's classifications and the subject's privileges which determine the types of MAC access allowed. Refer to Section 1.4 of this manual for information about this topic.

† This is why the $CHKPRO system service has two different items which refer to read and write access. The DAC access type is passed in the CHP$_ACCESS; the MAC access type is passed in CHP$_FLAGS.

1.6 Logging in

Users can be assigned a range of classifications by the system manager or security manager. However, a user process can operate at one classification at a time only. The classification of a session is specified when a user logs in by using the /SECRECY qualifier with the user name.

In the following example, user Jones logs in and specifies a classification of SYSTEM_LOW for this session:

```
Username: Jones /SECRECY=(LEVEL:SYSTEM_LOW)
Password:

 Welcome to VMS V5.2...
 .
 .
 .
```

If the user does not specify the /SECRECY qualifier when logging in, the session classification defaults to the user's highest authorized classification.

The user cannot specify a classification when logging in to a DECwindows session on a VAXstation. Users must always log in at their *default* classification. Furthermore, in this version of SEVMS, users are only allowed to log in to an *unclassified* DECwindows session (i.e. LEVEL = 0, CATEGORIES = NONE). Attempting to log in to a classified DECwindows session causes the following error message to be displayed:

```
Not Supported
```

If an attempt to log in fails for reasons related to mandatory access controls, the following message is returned:

```
User authorization failure
```

This message does not give the explicit reason for the rejection of the log-in. This prevents supplying any information to an unauthorized person who is making a determined attack on the system.

Some examples of reasons you may receive this message are:

- You attempted to log in at a classification for which you have no authorization.
- You incorrectly specified a classification.
- The classification that you requested is incompatible with the classification of the terminal you are attempting to log in on.
- You explicitly specified a classification when logging in to a CAPTIVE account.

1.7 Displaying Session Classification

At times, it may be necessary or useful to display the classification of the session you have logged in to. To display the classification of your session, use the SHOW CLASS/PROCESS command.

For example, you enter the command as follows:

```
$ SHOW CLASS/PROCESS
```

The type of information that is returned as a result of entering this command is illustrated in the following example.

```
Object type: process,  Object name: JONES,  on  13-MAY-1989 06:33:53.58
Class: SECRECY=(LEVEL=FIUO,CATEGORY=(NONE))
```

1.8 Classifying Directories and Files

This section describes the method of classifying directories and files.

Who Should Classify Directories and Files?

Depending upon the site, either the security manager, system manager, or the user may have the task of classifying each user's directories and files when SEVMS is installed.

When Should Directories and Files be Classified?

The classification of directories and files must be done when SEVMS is first installed on the system, or when new accounts are added on disks. In the case of an existing SEVMS system or account, SEVMS automatically classifies newly created files and directories with the same classification of the process creating the file or directory.

What is the Method of Classifying Directories and Files?

Classifying directories and files requires the following steps:

1 Determine the classification of each file to be classified.

2 Determine a directory structure for these files (i.e. how files should go into directories).

3 Organize the directories and files according to the way that you've determined they are to be classified.

4 Set the classification of the directories and files.

The above steps are described in detail in Section 1.8.1, Section 1.8.2, Section 1.8.3, and Section 1.8.4, which follow.

Note about Propagation of Classification

As mentioned above, when you are dealing with an already existing SEVMS system or account, newly created files and directories are given, by default, the classification of the process you are currently logged in at. Therefore, when you create a file or directory, it acquires the classification of your process, not the classification of the directory in which the file or directory file resides.

1.8.1 Determining File Classification

Examine the files and directories that you wish to classify. Determine which files should be classified at a particular secrecy level according to the needs of your site.

1.8.2 Determining Directory Structure

After determining how you are going to classify your files, you need to determine how these files are to be arranged in your directory (i.e. the directory structure).

To determine a directory structure for the files which you intend to classify, you need to establish the following:

- What directories should be created?
- What classification should each directory have?
- Which files should go into which directories?

After you have established the way you are going to organize your files and directories, you are ready to go on to the next step.

1.8.3 Organizing Directories and Files

When organizing directories and files, keep the following points in mind:

- Directories and files must be organized into a hierarchy that corresponds to the hierarchy of secrecy levels (i.e. lower level directories should have higher classifications than higher level directories).
- To access a file, your process must have *read access* to all the directories in the directory string.

These two points have the following important implication. **A directory should never have a subdirectory with a classification that is lower than the classification of the parent directory.**

To further clarify this idea, consider the following example.

Suppose that your account contained the following directory structure:

[HOTDOG.BASEBALL.JUNKFOOD]

These directories are classified as SYSTEM_LOW, TOP_SECRET, and SYSTEM_LOW, respectively.

```
        [HOTDOG] (system_low classification)
            |
            |
    [HOTDOG.BASEBALL] (top_secret classification)
            |
            |
[HOTDOG.BASEBALL.JUNKFOOD] (system_low classification)
```

In this example, if your current process is operating at SYSTEM_LOW, you would be unable to access the SYSTEM_LOW [...JUNKFOOD] directory, since your process does not have *read access* to the preceding TOP_SECRET [...BASEBALL] directory.

The proper way to organize the directory structure in the above example i to make the TOP_SECRET [...BASEBALL] directory a subdirectory to the SYSTEM_LOW [...JUNKFOOD] directory, as shown below:

[HOTDOG.JUNKFOOD.BASEBALL]

Now, the secrecy levels of these directories are organized in the following order: SYSTEM_LOW, SYSTEM_LOW, TOP_SECRET, respectively.

```
          [HOTDOG] (system_low classification)
              |
              |
      [HOTDOG.JUNKFOOD] (system_low classification)
              |
              |
  [HOTDOG.JUNKFOOD.BASEBALL] (top_secret classification)
```

With the directory structure organized in this way, if your current process is operating at SYSTEM_LOW, you can access both of the directories that are classified at SYSTEM_LOW.

As you go down in the directory structure, the secrecy level should increase. However, you can have different secrecy levels at the same level in the directory structure. For example, your log-in directory can have SYSTEM_LOW and CONFIDENTIAL subdirectories.

Therefore, the directories in the original example could also have been organized as follows:

[HOTDOG.JUNKFOOD]
[HOTDOG.BASEBALL]

In this case, the higher level directory, HOTDOG, is classified at SYSTEM_LOW; the subdirectory, JUNKFOOD, is classified at SYSTEM_LOW; and the subdirectory, BASEBALL, is classified at TOP_SECRET.

```
                          [HOTDOG] (system_low)
                              |
                              |
          -----------------------------------------
          |                                       |
[HOTDOG.JUNKFOOD] (system_low)        [HOTDOG.BASEBALL] (top_secret)
```

Figure 1-1 contains another example of a possible directory structure. In this example, the classification appears abbreviated after the directory name (SYSTEM_LOW is abbreviated to SL, TOP_SECRET to TS).

Note that [LOCAL.JUNKFOOD] and [LOCAL.PROJECT] exist at the same directory level, but have different classifications.

Figure 1-1 Directory Structure Example

```
                                 [LOCAL](SL)
                                    |
                                    |
                      -----------------------------
                      |                           |
                      |                           |
              [LOCAL.JUNKFOOD](SL)         [LOCAL.PROJECT](TS)
                      |                           |
                      |                           |
           -----------------------                |
           |                     |                |
           |                     |                |
[LOCAL.JUNKFOOD.A](TS)    [LOCAL.JUNKFOOD.B](SL)  |
                                                  |
                                         -------------------
                                         |                  |
                                         |                  |
                           [LOCAL.PROJECT.PLANS](TS)        |
                                                            |
                                                 [LOCAL.PROJECT.BOOK](TS)
```

1.8.4 Setting the Classification of Directories and Files

After organizing your directories and files, you are ready to set their classification. To classify a file or directory, you must have mandatory READ and WRITE access, and discretionary CONTROL access to the object. Also, keep in mind that if you classify a file with a classification that is not in the classification range of your account, you may need privileges to access that file.

Use the following order when setting the classification of directories and files:

1 Set the classification of the files within the directories.

2 Set the classification of each directory.

Classifying Files

To set the classification of a file, use the SET CLASS command. For example, to classify all the files in the default directory at FIUO, you would use the following command:

```
$ SET CLASS *.*;* /SECRECY=(LEVEL:FIUO)
```

Classifying Directories

To set a directory classification, use the SET CLASS command. For example, to set the classification of the directory [SOCRATES.PROJECT], you would use the following command:

```
$ SET CLASS /SECRECY=(LEVEL:TOP_SECRET) [SOCRATES]PROJECT.DIR
```

2 DCL Commands

This chapter contains descriptions of all SEVMS DCL commands. SEVMS DCL commands are existing VMS DCL commands which have been modified, and new DCL commands which have been added, to provide users with new functionality for the SEVMS environment.

ANALYZE/AUDIT

The ANALYZE/AUDIT command analyzes the contents (audit records) of an audit archive file. Only SEVMS-specific usage of this command is explained in this section. For a full description of ANALYZE/AUDIT, refer to the *VMS Audit Analysis Utility Manual* and the *Guide to VMS System Security*.

FORMAT

ANALYZE/AUDIT *file-spec[,...]*

restrictions

Refer to the *VMS Audit Analysis Utility Manual* and the *Guide to VMS System Security*.

PARAMETERS

file-spec[,...]

Specifies the name of the audit archive file. Refer to the *VMS Audit Analysis Utility Manual* and the *Guide to VMS System Security* for details.

DESCRIPTION

The ANALYZE/AUDIT command enables the user to analyze the audit records contained in an audit archive file.

The information provided in this section pertains only to the use of the ANALYZE/AUDIT command in relation to SEVMS. A full description of ANALYZE/AUDIT is provided in the *VMS Audit Analysis Utility Manual* and the *Guide to VMS System Security*.

For SEVMS, several keywords have been added to ANALYZE/AUDIT command qualifiers to enable auditing of mandatory access control events. The /SELECT qualifier uses the SEVMS keywords UPGRADE, DOWNGRADE, JOBNUM, and SYMBIONT_ID.

These keywords are described in detail in the following sections.

QUALIFIERS

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

The /SELECT qualifier specifies the criteria to be used when selecting event records. The following SEVMS-related criteria can be specified:

Keyword	Description
PRIVILEGES_USED=(privs[,...])	Specifies the privileges of the process to be used in selecting event records. The following privileges can be specified: UPGRADE DOWNGRADE

Keyword	Description
SYMBIONT_ID	Specifies the process ID for the print symbiont. A wildcard (*) is allowed.
JOBNUM	Specifies the number of the print job. A wildcard (*) is allowed.

EXAMPLES

1

```
$ANALYZE/AUDIT/FULL/SELECT=PRIVILEGES_USED=UPGRADE -
_$ SYS$MANAGER:SECURITY_AUDIT.AUDIT$JOURNAL
```

This example produces a report which selects all records written to the security audit log file that were generated by events through the use of the UPGRADE privilege.

2

```
$ANALYZE/AUDIT/SELECT=SYMBIONT_ID=150 AUDIT.LOG

    Date / Time          Type       Subtype       Node      Username     ID
-----------------------------------------------------------------------------
25-SEP-1989 18:18:29.75  PRINT     PRINT_FAIL    TURBOII   RUSNERS   00000150
25-SEP-1989 18:40:21.08  PRINT     PRINT_FAIL    TURBOII   RUSNERS   00000150
25-SEP-1989 18:46:27.36  PRINT     PRINT_FAIL    TURBOII   RUSNERS   00000150
25-SEP-1989 19:23:09.75  PRINT     PRINT_FAIL    TURBOII   RUSNERS   00000150
25-SEP-1989 19:37:55.73  PRINT     PRINT_FAIL    TURBOII   RUSNERS   00000150
26-SEP-1989 07:27:43.47  PRINT     PRINT_FAIL    TURBOII   RUSNERS   00000150
26-SEP-1989 07:36:01.49  PRINT     PRINT_SUCC    TURBOII   RUSNERS   00000150
```

This example produces a report which selects all records written to the security audit log file that have a symbiont ID number of 150.

3

```
$ANALYZE/AUDIT/FULL/SELECT=PRIVILEGES_USED=UPGRADE -
_$ SYS$MANAGER:SECURITY_AUDIT.AUDIT$JOURNAL
```

This example produces a report which selects all records written to the security audit log file that were generated by events through the use of the UPGRADE privilege.

4

```
$ANALYZE/AUDIT/SELECT=SYMBIONT_ID=* AUDIT.LOG
```

This example uses a wildcard to select all records which have a SYMBIONT_ID. This, in effect, produces a report which selects all records written to the security audit log file resulting from print events.

SET

The SET command defines or changes classifications associated with an object (SET CLASS), defines the templates to be associated with a category (or categories) of print jobs (SET TEMPLATE), and audits files according to classification (SET AUDIT). Equivalent commands for the SET CLASS command can be given using SET DEVICE, SET DIRECTORY, and SET FILE with SEVMS-specific qualifiers; however, for consistency, use of SET CLASS is recommended.

FORMAT

SET *option/SECRECY=string parameters*

restrictions

- The qualifier /SECRECY is mandatory for SEVMS SET commands.

PARAMETERS

See the individual SET command options.

DESCRIPTION

The SET command options for SEVMS are briefly described in the following table. A full description of each SET command option is included in the following pages of this section. The SEVMS SET commands augment the standard VMS SET commands.

Option	Function
AUDIT	Perform SEVMS auditing functions which allow access to classified files to be audited.
CLASS	Modify the classification of an object.
DEVICE	Modify the classification of a device.
DIRECTORY	Modify the classification of a directory.
FILE	Modify the classification of one or more files.
TEMPLATE	Associate a printed template to be output with all print jobs possessing certain attributes.

QUALIFIERS

/SECRECY

Used with SET AUDIT to indicate mandatory access control auditing. See the description of the SET AUDIT command for more information.

/SECRECY=class-string

The /SECRECY=class-string qualifier is used with the SET CLASS, SET DEVICE/CLASS, SET DIRECTORY/CLASS, and SET FILE/CLASS commands. This qualifier is not used with the SET AUDIT command (see the preceding /SECRECY qualifier description).

Note: The maximum length for a qualifier value in VMS Version 5.2 is 256 characters. If a class-string longer than this must be specified, use the /SQn qualifiers which are described in this section.

The /SECRECY=class-string qualifier specifies a secrecy level and/or categories, or a range of secrecy levels and/or categories. SECURITY privilege is needed to set a classification to a range. DOWNGRADE privilege is needed to lower a secrecy classification.

/SECRECY Qualifier Syntax

```
secrecy-qualifier := /SECRECY=(class-string)
                  ( LEVEL= level-range |                     )
                  ) CATEGORY= category-range |               (
class-string :=  {  LEVEL= level-range,                       }
                  (      CATEGORY= category-range            )
                 ( level |                                )
level-range :=  {  (MAXIMUM:level) |                       }
                 ( (MINIMUM:level, MAXIMUM:level)         )
                     ( category-list |                        )
                     ) (MAXIMUM:(category-list)) |            (
category-range :=   {  (MINIMUM:(category-list),               }
                     (      MAXIMUM:(category-list))          )
category-list :=  { category |               }
                  { (category [, . . . ])     }
```

Notes:

If not entered, minimum ranges default to 0 or none.
Colons (:) and equals signs (=) can be used interchangeably in the qualifier.
If MAXIMUM is specified for a single level object, it is ignored.

See the specific command for more information.

/INTEGRITY = class-string

All SET commands which take a /SECRECY=class-string qualifier also take an /INTEGRITY=class-string qualifier. This qualifier is used to set integrity classifications. This qualifier is not documented in the individual SET commands to simplify their descriptions - since integrity is seldom used. UPGRADE or BYPASS privilege is needed to raise an integrity classification.

/SQ1 = "class-string part-1"

/SQ2 = "class-string part-2"

/SQ3 = "class-string part-3"

/SQ4 = "class-string part-4"

The /SQn qualifiers are used to specify class strings which are longer than 256 characters. A class string which exceeds 256 characters can be broken up into 2, 3, or 4 parts, each less than 256 characters, by using one or more of these qualifiers. Each part must be enclosed in quotation marks (" "). The entire class string must be specified in a single command.

The format for the /SQn qualifier is:

SET *option* /**SQ1** = *"string parameters"* /**SQ2** = *"string parameters"* -
_/**SQ3** = *"string parameters"* /**SQ4** = *"string parameters"*

EXAMPLES

1

```
$SET CLASS/SQ1="(LEVEL=" -
_$ /SQ2="SECRET,CAT" -
_$ /SQ3="EGORY=RED)" FILE.DAT
```

This command qualifier enables the class string to be broken up into 3 parts.

SET AUDIT

The SET AUDIT command performs SEVMS auditing functions which enable auditing of file access according to classification.

FORMAT

SET AUDIT */SECRECY*
/ALARM

restrictions

- Requires SECURITY privilege.
- The /SECRECY qualifier cannot be used with the /ALARM qualifier.
- The /SECRECY qualifier must be used with the /CATEGORY or /LEVEL qualifiers (or both).
- The /SECRECY qualifier requires the use of either /ENABLE or /DISABLE qualifiers.

PARAMETERS *none*

DESCRIPTION

The /SECRECY qualifier, if specified, must be given without any value.

The /SECRECY qualifier, mandatory for controlling auditing by classification, is incompatible with the /ALARM qualifier, which is required on the standard VMS SET AUDIT command. These two qualifiers are mutually exclusive.

The /ENABLE or /DISABLE qualifier is also required. When used with /SECRECY, the event types FILE_ACCESS, PRINTED_FILE, CHANGE_CLASS, and LABEL_BYPASS are allowed, together with the event classes SUCCESS and FAILURE. When used with /ALARM, all the normal VMS event types may be used, and, in addition, the keywords DOWNGRADE and UPGRADE are available with event type FILE_ACCESS.

The result of the SET AUDIT command can be displayed using the DCL command SHOW AUDIT.

Note: SET AUDIT/SECRECY does not replace the SET AUDIT/ALARM command, but supplements it, allowing auditing by classification.

QUALIFIERS

/SECRECY

The /SECRECY qualifier causes auditing to be done by the secrecy classification of objects. This qualifier is required to control the auditing of file access, classification change, printing of files, and attempts to bypass page labeling when printing files by secrecy classification. To enable or disable the auditing by classification, both /SECRECY, /LEVEL and/or /CATEGORY, and either /ENABLE or /DISABLE are required.

/ALARM

SEVMS adds two keywords that can be used *only* with the /ENABLE=FILE_ACCESS and /DISABLE=FILE_ACCESS qualifiers of the SET AUDIT/ALARM command. The new keywords enable auditing of the use of DOWNGRADE or UPGRADE privilege to access a file. Note that they do *not* audit the use of SET CLASS to change the classification of a file.

The new keywords and the events for which they enable (or disable) alarms are:

New Keywords for FILE_ACCESS	**Events audited**
DOWNGRADE[:access[,access]]	Successful file access due to the use of the DOWNGRADE privilege
UPGRADE[:access[,access]]	Successful file access due to the use of the UPGRADE privilege

The DOWNGRADE privilege allows a user to write to a lower secrecy object; the UPGRADE privilege allows the user to write to a higher integrity object. See the SET AUDIT command description in the *VMS DCL Dictionary* for a description of SET AUDIT/ALARM and a list of the file_access keywords.

Note: **To display the status of these alarms use the SHOW AUDIT command, not SHOW AUDIT/SECRECY.**

/LEVEL=(level[,...])

Specifies the secrecy level or levels for which the specified event is, or is not, audited. An asterisk (*) can be used to indicate all levels.

One or more secrecy levels can be specified. Auditing is enabled or disabled for each level in the list; the list specifies a set of separate levels, not a range.

If the /SECRECY qualifier is used, then either /LEVEL, /CATEGORY, or both, must be used. You must specify /ENABLE or /DISABLE or both.

/CATEGORY=(category[,...])

Specifies the secrecy category or categories for which the specified event is, or is not, audited. An asterisk (*) can be used to indicate all categories.

One or more secrecy categories can be specified.

If the /SECRECY qualifier is used, then either /LEVEL, /CATEGORY, or both, must be used. You must specify /ENABLE, /DISABLE or both.

/DISABLE=events

Disables security auditing for the specified events. Events which can be disabled are the same as those which can be enabled. See the /ENABLE qualifier description for a list of the events to use with the /DISABLE qualifier.

/ENABLE=events

Enables security auditing for the specified events. Audit events consist of an event type and one or more event classes.

event-type:(event-class[,event-class])

The following event types can be specified when /ENABLE is used with /SECRECY:

event-type	Meaning
FILE_ACCESS	attempts to access files
PRINTED_FILE	attempts to print a file
LABEL_BYPASS	attempts to bypass the page labeling on a file by using the PRINT/PASSALL command
CHANGE_CLASS	attempts to use the $CHANGE_CLASS system service

The following event classes may be specified when /ENABLE is used with /SECRECY:

event-class	Meaning
SUCCESS	the attempt succeeded
FAILURE	the attempt failed

When /ENABLE is used with /ALARM, all the normal event types may be used (see SET AUDIT in the *VMS DCL Dictionary*). In addition, the new keywords DOWNGRADE and UPGRADE are available with event type FILE_ACCESS.

EXAMPLES

1

```
$SET AUDIT/SECRECY/LEVEL=(SECRET,TOP_SECRET)/CATEGORY=(SENSITIVE) -
_$ /ENABLE=FILE_ACCESS=(SUCCESS,FAILURE)
```

This command enables auditing for all accesses to any file that has a level of SECRET or TOP_SECRET, as well as to any file with category SENSITIVE.

2

```
$SET AUDIT/SECRECY/LEVEL=TOP_SECRET/DISABLE=FILE_ACCESS=(FAILURE)
```

This command disables auditing for the unsuccessful access of a file that is level TOP_SECRET.

3

```
$SET AUDIT/ALARM/ENABLE=FILE_ACCESS:(DOWNGRADE,UPGRADE)
```

This command enables auditing of file accesses that succeed due to the possession of upgrade or downgrade privileges.

SET CLASS

The SET CLASS command changes the classification of an object.

FORMAT

SET CLASS */SECRECY = class-string object-name*

restrictions

- The /SECRECY qualifier is required.

PARAMETERS

object-name

Specifies the object whose classification is being modified. Wildcards are allowed in the file names.

DESCRIPTION

If the object is not a file, the /OBJECT_TYPE qualifier is required. If the /OBJECT_TYPE qualifier is not specified, a file object type is assumed.

When using this command, it should be noted that SEVMS prevents changes in the classification of the following objects:

- File oriented devices (i.e. disk and tapes) with volumes mounted on them.
- Global sections backed by files that are mapped read/write.

QUALIFIERS

/SECRECY = class-string

See the description of this qualifier under the SET command.

/OBJECT_TYPE = type

Specifies the type of the object whose classification is being modified.

By default, a file object type is assumed. If the object is not a file, the /OBJECT_TYPE qualifier is required. The following keywords may be specified with the /OBJECT_TYPE qualifier:

Type	Meaning
FILE	Specifies that the object type is a file or a directory file.
DEVICE	Specifies that the object type is a device.
SYSTEM_GLOBAL_SECTION	Specifies that the object type is a system global section.
GROUP_GLOBAL_SECTION	Specifies that the object type is a group global section.
LOGICAL_NAME_TABLE	Specifies that the object type is a logical name table.

Type	Meaning
QUEUE	Specifies that the object type is a queue.

EXAMPLES

1
```
$SET CLASS/SECRECY=(LEVEL:SECRET)/OBJECT_TYPE=DEVICE  DLA0
```

The user changes the secrecy classification of the device DLA0 to a level of "SECRET".

2
```
$SET CLASS/SECRECY=(LEVEL:TOP_SECRET,CATEGORY=(RED)) FOO.BAR
```

The user changes the secrecy classification of the file FOO.BAR to level TOP_SECRET, category RED.

SET CLASS/NODE

The SET CLASS/NODE command associates classifications (or classification ranges) with various types of DECnet links for a specified node.

FORMAT

SET CLASS/NODE */LINK=(keyword) node-name*
/SECRECY=class-string

restrictions

- The /LINK qualifier is required.
- The /SECRECY, or /INTEGRITY, qualifier is required.
- SYSTEM privilege is required.
- SECURITY privilege is required.

PARAMETERS

node-name

Specifies the node to be associated with a classification.

DESCRIPTION

The SET CLASS/NODE command provides classification control of DECnet logical links on a node by node basis. It limits the establishment of links with the specified node to processes running within the specified classification range (or at a single specified classification). Different limitations can be set for different kinds of links. This command can also be used to specify a classification for links from a non-SEVMS node. A set of keywords for the /LINK qualifier are used to specify the type of connections that are affected by the command.

The limitations established by associating classifications with remote, incoming, and outgoing links are used in addition to any existing limitations established by other means. The SET CLASS/NODE command will not override other restrictions. For instance, if RTA0: is classified SECRET, remote logins will only be allowed at that classification. If, in addition, remote links from a node were classified TOP_SECRET, no remote logins would be allowed from that node at all.

The use of the SET CLASS/NODE command is optional, and only required if different behavior must be established for different nodes.

The classifications input with this command are stored in a system data file. Therefore, the command need not be executed each time the system is rebooted.

Note: **Refer to the *VMS SES Installation Guide and Release Notes* for further restrictions.**

This command does not check to verify the actual existance of the node being classified. This command requires SYSTEM privilege.

QUALIFIERS

/SECRECY = class-string

See the description of this qualifier under the SET command.

/LINK = (keyword)

The /LINK qualifier specifies the type of connections that are affected by the SET CLASS/NODE command.

There are four keywords that can be used with this qualifier: REMOTE, OUTGOING, INCOMING, NOCLASSIFICATION. These keywords are described in the following table.

The /LINK qualifier is required.

Link Type	Meaning
REMOTE	A SET HOST from the specified node can only log in within the indicated classification range. Other limitations on remote login continue to apply. (i.e. A non-privileged user can only log in at the same classification as the remote process.)
OUTGOING	A process shall succeed in requesting links to the specified node only if it is within the indicated classification range.
INCOMING	A process will receive incoming link requests from a remote process running on the specified node only if the remote process is running within the indicated classification range.
NOCLASSIFICATION	Overrides the default classification (unclassified) SEVMS uses for incoming connections that have no classification information. This includes connection requests from non-SEVMS nodes, as well as connections from unclassified processes running on SEVMS nodes. Because SEVMS cannot differentiate between the two cases, it is recommended that this keyword be used with non-SEVMS nodes only. A classification overridden by /NOCLASSIFICATION will be used in the check for /INCOMING and /REMOTE links.

EXAMPLES

❶
```
$SET CLASS/NODE/LINK=REMOTE TURBO/SECRECY=(LEVEL:SECRET)
```

A SET HOST from the node TURBO will only be allowed to login at SECRET.

❷
```
$SET CLASS/NODE/LINK=NOCLASSIFICATION TURBO/
SECRECY=(LEVEL:SECRET)
```

Any link requests from node TURBO without classification information will be treated as they were from SECRET processes. Link requests from non-SEVMS systems and link requests from unclassified processes on SEVMS systems have no classification. This command would not be recommended if TURBO were an SEVMS system.

SET CLASS/SERVER

The SET CLASS/SERVER command specifies the classification for connections from ports on a specified terminal server.

FORMAT

SET CLASS/SERVER *server-name*
/SECRECY = class-string

restrictions

- The /SECRECY or /INTEGRITY qualifier is required.
- The SYSPRV privilege is required.

PARAMETERS

server-name
Specifies the specific terminal server which is to be classified.

DESCRIPTION

The SET CLASS/SERVER command sets the classification which will be given to the LTAn: terminal device created when a connection is received from the specified terminal server. If the /PORT qualifier is specified, the classification will apply only to the specified port on the specified server. If the /PORT qualifier is not specified, the classification will apply to all ports on the terminal server.

The classifications input with this command are stored in a system data file; the command need **not** be executed each time the system is rebooted.

This command will only affect the classification assigned by the node or cluster it is issued on. If more than one SEVMS node or cluster is sharing a terminal server, the SET CLASS/SERVER command must be issued on each node or cluster.

This command does not check to verify the actual existance of the terminal server or ports being classified. This command requires the SYSPRV privilege.

QUALIFIERS

/SECRECY = class-string
See the description of this qualifier under the SET command.

/PORT = remote-port-name
The /PORT qualifier is used to set the classification of a specific port on a specified terminal server. When using this qualifier, specify the server-name parameter after the remote-port-name. An example which illustrates this is included in this section.

EXAMPLES

1
```
$SET CLASS/SERVER FLOOR3/SECRECY=(LEVEL:SECRET)
```

The user changes the secrecy classification of all terminal ports on LAT server FLOOR3 to level SECRET.

2
```
$SET CLASS/SERVER/PORT=LC-1-9 FLOOR3/SECRECY=(LEVEL:SECRET)
```

The user changes the secrecy classification of the terminal port LC-1-9 on server FLOOR3 to level SECRET.

SET DEVICE/CLASS

The SET DEVICE/CLASS command changes the classification of a device.

FORMAT

SET DEVICE/CLASS */SECRECY=class-string*
device-name[:]

restrictions

- The /SECRECY qualifier is required.
- The /CLASS and /SECRECY qualifiers cannot be used with the other standard VMS SET DEVICE qualifiers. The standard VMS SET DEVICE qualifiers are ignored if these qualifiers are used.

PARAMETERS

device-name
Specifies the name of the device whose classification is to change.

QUALIFIERS

/SECRECY=class-string
See the description of this qualifier under the SET command.

EXAMPLES

1
```
$SET DEVICE/CLASS/SECRECY=(LEVEL:0,CATEGORY:1) DUA0
```

The user replaces the classification of device DUA0 with the classification of secrecy level 0 and secrecy category 1.

Note that SET DEVICE/CLASS/SECRECY is equivalent to SET CLASS /OBJECT_TYPE=DEVICE /SECRECY. *For the sake of consistency, use of SET CLASS is recommended.*

SET DIRECTORY/CLASS

The SET DIRECTORY/CLASS command changes the classification of a directory.

FORMAT

SET DIRECTORY/CLASS */SECRECY=class-string directory-spec*

restrictions

- The /SECRECY qualifier is required.
- The /CLASS and /SECRECY qualifiers cannot be used with the other standard VMS SET DIRECTORY qualifiers. The standard VMS SET DIRECTORY qualifiers are ignored if these qualifiers are used.

PARAMETERS

directory-spec
Specifies a directory file to be modified.

QUALIFIERS

/SECRECY=class-string
See the description of this qualifier under the SET command.

EXAMPLES

1

```
$SET DIRECTORY/CLASS/SECRECY=(LEVEL:0,CATEGORY:3) [user]
```

The user replaces the classification of directory USER with the classification of secrecy level 0 and secrecy category 3.

Note that the SET DIRECTORY/CLASS/SECRECY [directory] is equivalent to SET CLASS/SECRECY [-]directory.dir. *For the sake of consistency, use of SET CLASS is recommended.*

SET FILE/CLASS

The SET FILE/CLASS command changes the classification of a file.

FORMAT

SET FILE/CLASS */SECRECY=class-string*
file-spec[,...]

restrictions

- The /SECRECY qualifier is required.
- The /CLASS and /SECRECY qualifiers cannot be used with the other standard VMS SET FILE qualifiers. The standard VMS SET FILE qualifiers are ignored if these qualifiers are used.

PARAMETERS

file-spec[,...]

Specifies one or more files whose classification is to change. If you specify two or more files, separate them with commas.

Wildcard characters are allowed in the file specifications.

QUALIFIERS

/SECRECY=class-string

See the description of this qualifier under the SET command.

EXAMPLES

1

```
$SET FILE/CLASS/SECRECY=(LEVEL:0,CATEGORY:1) USR:[TEST]A.TXT
```

The user replaces the classification of file A.TXT with the classification of secrecy level 0 and secrecy category 1.

Note that SET FILE/CLASS/SECRECY is equivalent to SET CLASS/SECRECY. *For the sake of consistency, use of SET CLASS is recommended.*

SET TEMPLATE

The SET TEMPLATE command allows you to add, remove, or replace the association of an SEVMS print symbiont template with a specified secrecy level, category mask, printer, and printer width.

It also can be used to create a new, empty, template database (SEVMS$SMB_HDRFRM.DAT) file.

FORMAT

SET TEMPLATE *[template-name]*

restrictions

- Requires the SYSPRV privilege.

PARAMETERS

template-name

The name of the template for which an association is to be added or replaced. The *template-name* is required if the /ADD or /REPLACE qualifiers are specified. It is not used if the /REMOVE or /CREATE_DATABASE qualifiers are specified.

DESCRIPTION

The SET TEMPLATE command is used to control which templates the SEVMS print symbiont uses to print files. It does so by maintaining associations between template names and selection criteria in the symbiont database. The template is selected by the secrecy classification of the file, the device the file is being printed on and the width of the device.

SEVMS print symbiont templates, their creation, selection and use are discussed in the *VMS SES Security Manager's Guide*.

The command takes two types of qualifiers. The *operation* qualifiers specify what type of operation is to be performed; /ADD is the default. The *association* qualifiers indicate what the template is to be associated with; at least one association qualifier must be specified unless the /CREATE_DATABASE qualifier is specified.

OPERATION QUALIFIERS

/ADD (default)

Specifies that a new association should be added for the template. If a template already has the same association specified, an error is returned.

/CREATE_DATABASE

Specifies a new, empty, template association database should be created.

/REMOVE

Specifies that an association should be removed from the database.

/REPLACE

Specifies that a new association should replace the existing association for the template. If no template had the same association, an error is returned.

ASSOCIATION QUALIFIERS

/CATEGORY_MASK=(secrecy-category[, . . .])
/CATEGORY_MASK=<ANY>

Specifies a secrecy category mask to associate with the template.

If this qualifier is not specified, or the special <ANY> keyword is used, the template will be associated with any secrecy category mask.

/LEVEL=secrecy-level
/LEVEL=<ANY>

Specifies a secrecy level to associate with the template.

If this qualifier is not specified, or the special <ANY> keyword is used, the template will be associated with any secrecy level.

/PRINTER=printer-name
/PRINTER=<ANY>

Specifies a printer to associate with the template. The printer-name can be either a physical device name (i.e. LPA0) or an SEVMS logical printer name. SET TEMPLATE does not check for the existence of the printer.

If this qualifier is not specified, or the special <ANY> keyword is used, the template will be associated with any printer.

An SEVMS logical printer name is established for the device ddcu: by defining the logical name SEVMS$SMB_PRINTER_ddcu in the SYSTEM logical name table. The print symbiont will then use its definition rather than the physical device name to look for template associations. Refer to Chapter 7 of the *VMS SES Security Manager's Guide* for more information.

Note: **The <ANY> keyword is provided for the qualifiers above to be consistent with the SHOW TEMPLATE command. Its use is optional, since the same effect on association can be obtained by omitting the qualifier.**

/WIDTH=n

Specifies a printer line width to associate with the template.

If this qualifier is not specified, the template will be associated with a width of 0.

EXAMPLES

❶
```
$ SET TEMPLATE/CATEGORY=(1) CONFIDENTIAL_FORMAT
```

The print symbiont template named CONFIDENTIAL_FORMAT is associated with secrecy category 1.

2

```
$ SET TEMPLATE/PRINTER=<ANY>/WIDTH=80 C80
```

This associates the print symbiont template named C80 with printer width 80. The */PRINTER=<ANY>* qualifier is not needed since the default is to associate the template with any printer.

3

```
$ SET TEMPLATE/LEVEL=SECRET/CATEGORY=(GREEN,YELLOW)/PRINTER=LINE -
_$ /WIDTH=132 EXAMPLE_111
```

This command associates the template named EXAMPLE_111 with secrecy level SECRET, category mask GREEN and YELLOW, SEVMS logical printer LINE, and printer width 132.

4

```
$ SET TEMPLATE/CATEGORY=(GREEN,YELLOW) EXAMPLE_222
```

This command associates the template named EXAMPLE_222 with category mask GREEN and YELLOW.

5

```
$ SET TEMPLATE/CATEGORY=(GREEN,YELLOW)/REMOVE
```

This command removes the association between the secrecy category mask of GREEN and YELLOW and the SEVMS print symbiont template associated with it (if any). This command does not affect any other print symbiont template except those with an exact match. (In this case, the template named EXAMPLE_222 is matched, and therefore the association is removed. Note that the association for the template named EXAMPLE_111 is unaffected.)

SHOW AUDIT

The SHOW AUDIT command displays the alarms that have been enabled with the SET AUDIT/SECRECY and SET AUDIT/INTEGRITY commands.

FORMAT

SHOW AUDIT ***/SECRECY***

restrictions

- Requires the SECURITY system privilege.
- The /SECRECY qualifier is required.

PARAMETERS *none*

DESCRIPTION

The SHOW AUDIT command requires the use of the /SECRECY qualifier for SEVMS.

The SHOW AUDIT/SECRECY command provides a display that identifies which security auditing features have been enabled with the SET AUDIT/SECRECY and SET AUDIT/INTEGRITY commands and the events that will be audited. There is not a SHOW AUDIT/INTEGRITY command; SHOW AUDIT/SECRECY is used to display both secrecy and integrity alarms.

This command is useful for checking which auditing features are enabled whenever you plan to add or delete features with a SET AUDIT/SECRECY or SET AUDIT/INTEGRITY command.

QUALIFIERS

/SECRECY

The /SECRECY qualifier causes the set of auditing features that were enabled with the SET AUDIT/SECRECY and SET AUDIT/INTEGRITY commands, and the events they report, to be displayed. This qualifier is required.

EXAMPLES

1

```
$SHOW AUDIT/SECRECY
Mandatory access alarms currently disabled
```

In this example, mandatory access alarms are disabled.

```
2     $SHOW AUDIT/SECRECY
      Mandatory access alarms currently enabled for:

      FILE ACCESS  SECRECY  FAILURE LEVEL: CONFIDENTIAL,SECRET,VERY_SECRET

                                 CATEGORY: RED

                           SUCCESS LEVEL:   CONFIDENTIAL, SECRET

                                 CATEGORY: RED, BLUE
```

This example displays classification-related audit information for FILE_ ACCESS. (LABEL_BYPASS and PRINTED_FILE information would also be displayed in this example if they were enabled.)

SHOW CLASS

The SHOW CLASS command allows you to view the classification associated with an object, process, LAT server, terminal port, or node.

FORMAT

SHOW CLASS *name*

restrictions

- Either the /OBJECT_TYPE, /PROCESS, or /SERVER qualifier is required if the *name* is not a file. These qualifiers are mutually exclusive.
- Wildcards are not allowed in the *name*.
- The /PORT qualifier cannot be used without the /SERVER qualifier.

PARAMETERS

name

Specifies the type of classification to be viewed. Wildcards are not allowed.

DESCRIPTION

This command enables the user to view the classification associated with an object, process, LAT server, terminal port, or node. The default of this command is an object which is a file; therefore, if a qualifier is not specified, a file object type is assumed for *name*. For all objects which are not files, the /OBJECT_TYPE qualifier is required. For any other type of *name*, a qualifier must be specified.

QUALIFIERS

/OBJECT_TYPE = type object-name

Specifies the type of the object whose classification is being viewed.

By default, a file object type is assumed. If the object is not a file, this qualifier is required. The following keywords may be specified with /OBJECT_TYPE:

Type	Function
FILE	Specifies that the object type is a file or a directory file.
DEVICE	Specifies that the object type is a device.
SYSTEM_GLOBAL_SECTION	Specifies that the object type is a system global section.
GROUP_GLOBAL_SECTION	Specifies that the object type is a group global section.
QUEUE	Specifies that the object type is a queue.

Type	Function
LOGICAL_NAME_TABLE	Specifies that the object type is a logical name table.

/PROCESS [process-name]

Displays the classification of your process or any current subprocess. The default is the current process. Only processes in your own group can be identified by name. To show the classification of a process outside your group you must use the /IDENTIFICATION qualifier.

Showing another process' classification requires:

- GROUP privilege to show other processes in the same group.
- BYPASS privilege to show processes more classified than your own.

/PROCESS/IDENTIFICATION = pid

Displays the classification of the process with the specified process ID. PID is the process identification number which you can get with the DCL SHOW SYSTEM command.

Showing another process' classification requires:

- GROUP privilege to show other processes in the same group.
- WORLD privilege to show processes outside your group.
- BYPASS privilege to show processes more classified than your own.

/SERVER server-name

This qualifier displays the classification of a LAT server. The server-name parameter specifies the name of the terminal server which is to have it's classification displayed.

Using this qualifier to show the classification of a LAT server requires SYSPRV privilege.

If the /SERVER qualifier is used alone, it shows only the default classification (if any) for all ports on a specified server. To show the classification of individual (specific) ports, the /PORT qualifier must also be used.

The use of * is allowed; it is used to list all servers.

/PORT = remote-port-name

This qualifier displays the classification of a port on a LAT server. The remote-port-name parameter specifies the name of a terminal port on a specified LAT server which is to have it's classification displayed.

This qualifier can only be used in conjunction with the /SERVER qualifier. Using this qualifier to show the classification of a terminal port requires SYSPRV privilege.

The use of * is allowed; it is used to list all ports.

/NODE/LINK = (keyword) node-name

Displays the classification(s) of the logical link(s) on the specified node.

The /LINK qualifier specifies the type of logical link classification to be displayed. Using this command qualifier requires the SYSPRV privilege.

The following keywords can be specified for the /LINK qualifier: REMOTE, INCOMING, OUTGOING, NOCLASSIFICATION, ALL.

The use of * is allowed; it is used to list all nodes.

Refer to the SET CLASS/NODE command for information about classifying nodes and a description of the /LINK keywords.

EXAMPLES

❶
```
$SHOW CLASS/OBJECT_TYPE=DEVICE  DUA0

Object type: device,
Object name: DUA0, on 20-MAY-1989 20:17:51.78

Class: SECRECY=(LEVEL=UNCLASSIFIED,CATEGORY=(NONE))
```

This example shows the classification of the device DUA0.

❷
```
$SHOW CLASS/PROCESS

Object type: process,
Object name: SMITH, on 20-MAY-1989 20:17:53.17

Class: SECRECY=(LEVEL=UNCLASSIFIED,CATEGORY=(NONE))
```

This example shows the classification of the current process.

❸
```
$SHOW CLASS/SERVER/PORT=* *

Server/port classifications on 30-MAY-1989 12:53:39.39

Server: FLOOR2
Class: SECRECY=(LEVEL=(MINIMUM=0,MAXIMUM=3),CATEGORY=(123))

Server: FLOOR3
Class: SECRECY=(LEVEL=0,CATEGORY=(NONE))

Server/Port: FLOOR3/LC-1-9
Class: SECRECY=(LEVEL=0,CATEGORY=(NONE))
```

This example shows the classifications of all ports on all servers.

❹
```
$SHOW CLASS/SERVER *

Server classifications on 30-MAY-1989 12:54:39.59

Server: FLOOR2
Class: SECRECY=(LEVEL=(MINIMUM=0,MAXIMUM=3),CATEGORY=(123))

Server: FLOOR3
Class: SECRECY=(LEVEL=0,CATEGORY=(NONE))
```

This example shows the classification of all servers.

❺
```
$SHOW CLASS/SERVER/PORT=* FLOOR3

Server/port classifications on 30-MAY-1989 12:54:05.79

Server/Port: FLOOR3/LC-1-9
Class: SECRECY=(LEVEL=0,CATEGORY=(NONE))
```

This example shows the classifications of all ports on the specified server.

6
```
$SHOW CLASS/SERVER FLOOR3

Server classifications on 30-MAY-1989 12:54:25.17

Server: FLOOR3
Class: SECRECY=(LEVEL=0,CATEGORY=(NONE))
```

This example shows the classification of a specified server.

7
```
$SHOW CLASS/NODE/LINK=NOCLASSIFICATION TURBO

    Node classifications on 21-SEP-1989 06:17:17.04

    Link Type: NOCLASSIFICATION    Node: TURBO
    Class: SECRECY=(LEVEL=UNCLASSIFIED,CATEGORY=(NONE))
```

This example shows the classification that is associated with any link requests from node TURBO that do not have a classification.

8
```
$SHOW CLASS/NODE/LINK=INCOMING TURBO

    Node classifications on 21-SEP-1989 06:17:37.97

    Link Type: INCOMING    Node: TURBO
    Class: SECRECY=(LEVEL=CONFIDENTIAL,CATEGORY=(NONE))
```

This example shows the classification that is required of incoming link requests on node TURBO.

9
```
$SHOW CLASS/NODE/LINK=ALL TURBO

Node classifications on  8-NOV-1989 14:00:44.84

Link Type : REMOTE    Node : TURBO
Class: SECRECY=(LEVEL=2,CATEGORY=(NONE))

Link Type : INCOMING  Node : TURBO
Class: SECRECY=(LEVEL=(MINIMUM=0,MAXIMUM=20),CATEGORY=(NONE))

Link Type : OUTGOING  Node : TURBO
Class: SECRECY=(LEVEL=(MINIMUM=0,MAXIMUM=10),CATEGORY=(NONE))

Link Type : NOCLASSIFICATION  Node : TURBO
Class: NO CLASSIFICATION FOUND
```

This example shows the classification of all links on node TURBO.

SHOW TEMPLATE

The SHOW TEMPLATE command displays the currently established associations between SEVMS print symbiont templates and secrecy levels, category masks, printers and printer widths.

FORMAT

SHOW TEMPLATE *[template-name]*

restrictions

- Requires the SYSPRV privilege.

PARAMETERS

template-name

The name of the template for which an association is to be viewed.

DESCRIPTION

The SHOW TEMPLATE command displays the template associations used by the SEVMS print symbiont. Either some or all associations can be shown, as determined by the parameters and qualifiers specified, as follows:

- If no template name or qualifiers are specified, then all associations a shown.
- If only a template name is specified, then only associations for the template will be shown.
- If only qualifiers are specified, then only associations that match the qualifiers will be shown.
- If a template name and qualifiers are specified, then only associations that match the qualifiers for the template will be shown.

A special keyword, <ANY>, is provided to indicate that any value of a qualifier can be matched.

The creation, selection, and use of SEVMS print symbiont templates is discussed in the *VMS SES Security Manager's Guide.*

QUALIFIERS

/CATEGORY_MASK=(secrecy-category[, . . .])
/CATEGORY_MASK=<ANY>

Specifies a secrecy category mask that associations must match in order to be shown.

/LEVEL=secrecy-level
/LEVEL=<ANY>

Specifies a secrecy level that associations must match in order to be shown

/PRINTER=printer-name
/PRINTER=<ANY>

Specifies a physical or SEVMS logical printer name that associations must match in order to be shown.

/WIDTH=n

Specifies a printer width that associations must match in order to be shown.

EXAMPLES

1

```
$ SHOW TEMPLATE




Template associations in SYS$SYSROOT:[SYSLIB]SEVMS$SMB_HDRFRM.DAT;
at 2-AUG-1989 13:26:39.46

Printer: <ANY>, Width:   0, Level: CONFIDENTIAL,
    Category mask: <ANY>,
        Template: FOO

Printer: <ANY>, Width: 132, Level: <ANY>,
    Category mask: <ANY>,
        Template: BAR

Printer: <ANY>, Width: 132, Level: <ANY>,
    Category mask: <ANY>,
        Template: FOO
```

All template associations are shown.

2

```
$ SHOW TEMPLATE FOO




Template associations in SYS$SYSROOT:[SYSLIB]SEVMS$SMB_HDRFRM.DAT;
at 2-AUG-1989 13:26:51.14

Printer: <ANY>, Width:   0, Level: CONFIDENTIAL,
    Category mask: <ANY>,
        Template: FOO

Printer: <ANY>, Width: 132, Level: <ANY>,
    Category mask: <ANY>,
        Template: FOO
```

This command shows all associations for print symbiont template FOO.

3

```
$ SHOW TEMPLATE/LEVEL=<ANY>




Template associations in SYS$SYSROOT:[SYSLIB]SEVMS$SMB_HDRFRM.DAT;
at 2-AUG-1989 13:27:14.22

Printer: <ANY>, Width: 132, Level: <ANY>,
    Category mask: <ANY>,
        Template: BAR
```

```
Printer: <ANY>, Width: 132, Level: <ANY>,
    Category mask: <ANY>,
        Template: FOO
```

This command displays the templates that are not associated with a particular level.

```
4    $ SHOW TEMPLATE/WIDTH=132 FOO
```

```
Template associations in SYS$SYSROOT:[SYSLIB]SEVMS$SMB_HDRFRM.DAT;
at 2-AUG-1989 13:28:05.75

Printer: <ANY>, Width: 132, Level: <ANY>,
    Category mask: <ANY>,
        Template: FOO
```

This command displays only templates named FOO that were associated with a printer width of 132.

DIRECTORY

The DIRECTORY command provides a list of files, or information about a file or group of files.

FORMAT

DIRECTORY *[file-spec[,....]]*

PARAMETERS

[file-spec[,...]]

Specifies one or more files for which directory information is desired.

DESCRIPTION

Usage of the DIRECTORY command under SEVMS is similar to its usage under standard VMS. The only difference is that under SEVMS the classification of files is displayed if the /FULL or /SECURITY qualifiers are specified. Both of these qualifiers are available in VMS. Refer to the *VMS DCL Dictionary* for complete information on the DIRECTORY command.

QUALIFIERS

/FULL

Under SEVMS, the /FULL switch displays the classification of a file along with the other information displayed by the standard version of the DIRECTORY command.

/SECURITY

Under SEVMS, the /SECURITY qualifier displays the classification of a file along with the other security information displayed by the standard version of the DIRECTORY command.

EXAMPLES

1

```
$DIRECTORY/SECURITY [.DEMO]

Directory $DISK1:[J_SMITH.DEMO]

SECRET.DAT;1           [AGROUP,J_SMITH]        (RWED,RWED,RE,)
            SECRECY=(LEVEL=SECRET,CATEGORY=(NONE))
SECRET.MISC;1          [AGROUP,J_SMITH]        (RWED,RWED,RW,R)
            SECRECY=(LEVEL=SECRET,CATEGORY=(NONE))
SECRET_RED.MISC;1      [AGROUP,J_SMITH]        (RWED,RWED,RW,R)
            (ALARM_JOURNAL=SECURITY,ACCESS=READ+FAILURE)
            (IDENTIFIER=[123,456],ACCESS=READ+WRITE+EXECUTE)
            (IDENTIFIER=[SYSTEM],ACCESS=DELETE)
            SECRECY=(LEVEL=SECRET,CATEGORY=(RED))
SECRET_RED.TXT;1       [SYSTEM]                (RWED,RWED,RE,)
            SECRECY=(LEVEL=SECRET,CATEGORY=(RED))
UNCLASS.MISC;1         [AGROUP,J_SMITH]        (RWED,RWED,RW,R)
            (IDENTIFIER=[AGROUP,JONES],ACCESS=NONE)
            SECRECY=(LEVEL=UNCLASSIFIED,CATEGORY=(NONE))
UNCLASS.TXT;1          [SYSTEM]                (RWED,RWED,RE,)
            SECRECY=(LEVEL=UNCLASSIFIED,CATEGORY=(RED,63))

Total of 6 files.
```

This example displays the classification of all files in directory [J_SMITH.DEMO].

2

```
$DIRECTORY/FULL [.DEMO]UNCLASS.TXT

Directory $DISK1:[J_SMITH.DEMO]UNCLASS.TXT

UNCLASS.TXT;1                    File ID:  (869,20,0)
Size:             9/9          Owner:    [SYSTEM]
Created:  16-OCT-1989 13:23:10.86
Revised:  31-OCT-1989 08:13:44.84 (3)
Expires:   <None specified>
Backup:    <No backup recorded>
File organization:  Sequential
File attributes:    Allocation: 9, Extend: 0, Global buffer count: 0, No version limit
Record format:      Variable length, maximum 79 bytes
Record attributes:  Carriage return carriage control
RMS attributes:     None
Journaling enabled: None
File protection:    System:RWED, Owner:RWED, Group:RE, World:
Access Cntrl List:  None
Classification:     SECRECY=(LEVEL=UNCLASSIFIED,CATEGORY=(RED,63))

Total of 1 file, 9/9 blocks.
```

This example displays full information, which includes the classification, of the UNCLASS.TXT file in the directory [J_SMITH.DEMO].

INITIALIZE

The INITIALIZE command initializes a FILES-11 volume with the specified classification range.

FORMAT

INITIALIZE */SECRECY = class-range device-name volume-label*

PARAMETERS

device-name

Specifies the name of the device on which the volume to be initialized is physically mounted.

volume-label

Specifies the identification to be encoded on the volume.

DESCRIPTION

This command initializes a FILES-11 volume with the specified classification range. Standard INITIALIZE qualifiers can also be specified. See the INITIALIZE command in the *DCL Dictionary* for other qualifiers.

If you do not specify the /SECRECY qualifier, the volume classification defaults to the user's authorized classification range (specified in the UAF).

QUALIFIERS

/SECRECY = class-string

Specifies one or more secrecy levels and/or categories, or ranges of secrecy levels and/or categories.

/SECRECY Qualifier Syntax

```
secrecy-qualifier : = /SECRECY = (class-string)
                  ⎧ LEVEL = level-range |                 ⎫
class-string : =  ⎨ CATEGORY = category-range |           ⎬
                  ⎪ LEVEL = level-range,                  ⎪
                  ⎩       CATEGORY = category-range       ⎭
                 ⎧ level |                           ⎫
level-range : =  ⎨ (MAXIMUM:level) |                 ⎬
                 ⎩ (MINIMUM:level, MAXIMUM:level)    ⎭
                    ⎧ category-list |                  ⎫
category-range : =  ⎨ (MAXIMUM:(category-list)) |      ⎬
                    ⎪ (MINIMUM:(category-list),        ⎪
                    ⎩      MAXIMUM:(category-list))    ⎭
category-list : = ⎧ category |            ⎫
                  ⎩ (category [, . . . ]) ⎭
```

Notes:

If not entered, minimum ranges default to 0 or none.
Colons (:) and equals signs (=) can be used interchangeably in the qualifier.
If MAXIMUM is specified for a single level object, it is ignored.

/INTEGRITY = class-string

Specifies one or more integrity levels and/or categories, or ranges of integrity levels and/or categories.

EXAMPLES

1

```
$INITIALIZE/SECRECY=(LEVEL:2) DLO USER
```

This example initializes the disk which is mounted on DL0 and labeled USER with the classification of secrecy level 2.

3 Programming Information

This chapter provides programming information about SEVMS. The topics included in this chapter are: an explanation of the SEVMS class block format, descriptions of the system service routines that are employed by SEVMS, and a description of SEVMS manadatory access control items and arguments which are used with VMS System Services. This chapter is divided into the following sections:

- Class Block Format
- System Services
- SEVMS Binary Audit Record Format

3.1 Class Block Formats

There are two possible formats for the SEVMS class block. One format (Type 0) is used when only secrecy categories of 64 or less are specified; the other format (Type 1) is used when secrecy categories above 64 are specified.

The first class block format (Type 0) has the original standard SEVMS format. This format is used for class blocks that do not contain secrecy categories greater than 64 and may (or may not) have integrity categories. The following figure, Figure 3-1, illustrates this class block.

The second class block (Type 1) uses a slightly modified form of the standard SEVMS class block format. This format is used for class blocks that do contain secrecy categories greater than 64 and do not have integrity categories. The following figure, Figure 3-2, illustrates this class block.

3.2 System Services

This section contains complete reference descriptions of the system service routines used by SEVMS. It also includes information about the mandatory access controls support for new items in VMS system services.

3.2.1 SEVMS Mandatory Access Control Items in VMS System Services

This section describes new items used by VMS system service routines which are related to mandatory access controls.

Refer to the *VMS System Services* manual for further information about the system services discussed in this section.

3.2.1.1 $CHKPRO System Service

This system service can take the following items in its item list which are related to mandatory access controls:

- CHP$_ACCLASS—accessors classification
- CHP$_MINCLASS—minimum classification for the object
- CHP$_MAXCLASS—maximum classification for the object

3.2.1.2 $GETUAI System Service

This system service can take the following additional items in its item list which are related to SEVMS mandatory access controls:

- UAI$_MIN_CLASS—minimum class block for UAF record
- UAI$_MAX_CLASS—maximum class block for UAF record

3.2.1.3 $CREPRC System Service

The $CREPRC system service creates a subprocess or detached process on behalf of the calling process.

To support mandatory access controls, the argument *itmlst* can now be specified with the $CREPRC system service. The *itmlst* argument allows the classification of a detached process to be specified.

The use of this argument is optional. If this argument is not specified, the detached process will have the same classification as that of the creating process. If this argument is specified, the creating process must have BYPASS privilege *and* either UPGRADE (to change integrity) or DOWNGRADE (to change secrecy) privilege, in order to specify a classification for the detached process which is different than its creator's classification.

The *itmlst* argument has been implemented as the 13th positional argument of the $CREPRC system service.

ARGUMENT: ***itmlst***

VMS Usage: **item_list_2**
type: **longword (unsigned)**
access: **read only**
mechanism: **by reference**

The *itmlst* argument is the address of an item list of descriptors used to specify the classification of a detached process.

For each item code, include an item descriptor and terminate the list with a longword containing the value of 0. For a description of *item_list_2,* refer to the section on data types in the *Introduction to VMS System Routines* manual.

The following item code is used with $CREPRC.

Item Code	Description
6	The component address of a 20-byte classification block for a detached process. The component length must be 20.

RETURN VALUES:

The following status messages are returned if the classification of the detached process is different than that of the creating process and the creating process does not have the proper privileges.

SS$_NOUPGRADE
SS$_NODOWNGRADE
SS$_NOBYPASS

Note: **For this version of SEVMS, the value of the item code is 6. This value may change, so it should be assigned symbolically in user written programs. A standard VMS mnemonic will be assigned at a future date.**

3.2.1.4 $QIO System Service — ACP-QIO Interface

The $QIO system service can be used to request ACP operations by means of the ACP-QIO interface. One type of operation that can be requested by this interface is read/write attributes, to read or set the attributes of a file. The attributes to be read or set are specified by an attribute list.

One of the attributes that can be specified in the attribute list is the ATR$C_CLASS_MASK. The specification of this attribute allows reading, or setting of, a file's classification. (Note: In order to set a file's classification, the file must be closed.)

The ACP-QIO interface, including the format of the attribute list, is described in detail in the *VMS I/O User's Reference Manual.* Note that the ATR$C_CLASS_MASK is not documented in this manual as one of the attributes available. However, it is included in the system libraries that define these attributes (i.e. SYS$LIBRARY:STARLET.REQ), and is supported by SEVMS.

3.2.2 SEVMS System Service Routines

This section describes the system service routines provided by SEVMS.

3.2.2.1 Introduction

SEVMS provides three system service routines that can be used to manipulate classification labels. These routines are listed in the following table. Refer to the routine description section of this chapter for complete information about these system service routines.

Service	Description
$PARSE_CLASS	Converts ASCII text classification strings into a binary classification block.
$FORMAT_CLASS	Converts an internal binary classification block into ASCII text classification strings.
$CHANGE_CLASS	Gets and/or changes the classification of objects and processes.

3.2.2.2 Item Lists

All three system services take an item list argument whose items are identified by identical item codes, although a given system service may not use all the items. The type of item list used is an *item_list_3.* For a description of *item_list_3,* please refer to the section on data types in the *Introduction to VMS System Routines* manual. The following table, Table 3-1, describes the system service items used by SEVMS.

Table 3-1 SEVMS System Service Items

Item Code	Description
	Input Items
CLS$_SECSTR	The address of a text string descriptor which points to an ASCII secrecy classification string.[1,5]
CLS$_INTSTR	The address of a text string descriptor which points to an ASCII integrity classification string.[1,5]
CLS$_CLSBLK	The address of a descriptor which points to a 20 byte binary classification block.[2,4]
CLS$_MINCLSBLK	The address of a descriptor which points to a 20 byte binary classification block.[2,4]This block should contain the minimum classification for a ranged object. It has the same format as CLS$_CLSBLK.
CLS$_MAXCLSBLK	The address of a descriptor which points to a 20 byte binary classification block.[2,4]This block should contain the maximum classification for a ranged object. It has the same format as CLS$_CLSBLK.
CLS$_WIDTH	The address of a longword containing the maximum width, in characters, for the classification string produced by $FORMAT_CLASS.
CLS$_INDENT	The address of a longword containing the number of spaces to indent, with spaces, each line of the classification string produced by $FORMAT_CLASS.

[1] The item "buffer length" word should contain the length of the string or buffer, **NOT** the length of descriptor.

[2] The item "buffer length" word should contain 20, length of the classification block, **NOT** the length of the descriptor.

[4] Class blocks are defined in SYS$LIBRARY:LIB.MLB (for MACRO, LIB.REQ or LIB.L32 for BLISS) in the $CLSDEF macro.

[5] The maximum possible classification string length is 3072 bytes.

Table 3-1 (Cont.) SEVMS System Service Items

Item Code	Description
Input Items	
CLS$_TRMDSC	The address of a string descriptor which points to a character string that is used to terminate each line of the classification string produced by $FORMAT_CLASS.
Output Items	
CLS$_SECSTR	The address of a descriptor which points to a buffer that will contain an ASCII secrecy classification string.[3,5]
CLS$_INTSTR	The address of a descriptor which points to a buffer that will contain an ASCII integrity classification string.[3,5]
CLS$_OLDCLSBLK	The address of a descriptor which points to a 20 byte binary classification block that will contain the old classification of the object. It has the same format as CLS$_CLSBLK. [3,4]
CLS$_OLDMINCLS	The address of a descriptor which points to a 20 byte binary classification block that will contain the old minimum classification of the object. It has the same format as CLS$_CLSBLK. [3,4]
CLS$_OLDMAXCLS	The address of a descriptor which points to a 20 byte binary classification block that will contain the old maximum classification of the object. It has the same format as CLS$_CLSBLK. [3,4]

[3] The descriptor must be fixed length.

[4] Class blocks are defined in SYS$LIBRARY:LIB.MLB (for MACRO, LIB.REQ or LIB.L32 for BLISS) in the $CLSDEF macro.

[5] The maximum possible classification string length is 3072 bytes.

The item codes are defined in the system macro library ($CLSDEF).

$PARSE_CLASS

The PARSE CLASS service parses the specified classification text string and translates it to the corresponding binary classification block.

FORMAT

SYS$PARSE_CLASS *itmlst*

RETURNS

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

Longword condition value. All system services return (by immediate value) a condition value in R0. Condition values that can be returned by this service are listed under ''Return Values''.

ARGUMENTS

itmlst

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

The *itmlst* argument is the address of an item list of descriptors used to specify the mandatory access control protection attributes of an object.

For each item code, include an item descriptor and terminate the list with a longword containing the value of 0. For a description of *item_list_3*, refer to the section on data types in the *Introduction to VMS System Routines* manual.

The following item codes are used with $PARSE_CLASS. See Table 3-1 for information on the format of items.

Item Code	Use
CLS$_SECSTR	The secrecy classification string to be parsed.
CLS$_INTSTR	The integrity classification string to be parsed.
CLS$_CLSBLK	The resulting classification block.
CLS$_MINCLSBLK	The resulting minimum classification block.
CLS$_MAXCLSBLK	The resulting maximum classification block.

The text strings for the secrecy and integrity class have a similar format. These text strings may specify the minimum and maximum levels and applicable categories (or compartments) for the secrecy or integrity class.

The rules used to parse these text strings are:

- The initial keyword (SECRECY or INTEGRITY) may be omitted. This is to allow the text string to be handled easily as a value for a qualifier or value for a command.

- The values specified for the levels and categories may be identifiers (for example, SECRET) or a simple numeric value (for example, 4).
- The level may be a single value or a list specifying minimum and maximum values.
- The categories may be a single value or a list of values or lists of minimum and maximum values.
- If the text string only supplies a single level, but the item list requests minimum and maximum classification blocks, the resulting minimum and maximum classification blocks are the same.
- If the text string specifies that minimum and maximum classification blocks are to be generated, but only a single output block is specified in the item list, the minimum classification block is returned.

DESCRIPTION

$PARSE_CLASS converts a text string to the corresponding binary classification block.

- There is no way to determine from the output of $PARSE_CLASS whether level or category was specified in the input string. In other words the output from (LEVEL=0), (CATEGORY=0), and (LEVEL=0,CATEGORY=0) all look the same.
- The output classification block always has both the secrecy and integrity fields filled in; the fields are just 0 if no classification of a given type was specified on input. In other words, the output is the same whether a secrecy of (LEVEL=0) is input, or whether no secrecy string is input.

RETURN VALUES

SS$_NORMAL	The service successfully completed.
SS$_ACCVIO	The input string or its descriptor cannot be read, the output buffer descriptor cannot be read, the output buffer cannot be written, or the output buffer is not large enough to contain the binary classification block.
SS$_INVSECLASS	The input text string could not be correctly parsed to yield a binary secrecy class.
SS$_NOSUCHID	An identifier used to represent a secrecy level, secrecy category, integrity level, or an integrity category did not exist in the rights database.

$FORMAT_CLASS

The FORMAT CLASS service formats the specified binary classification block and converts it to the corresponding text string.

FORMAT

SYS$FORMAT_CLASS *itmlst*

RETURNS

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

Longword condition value. All system services return (by immediate value) a condition value in R0. Condition values that can be returned by this service are listed under "Return Values".

ARGUMENTS

itmlst

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

The *itmlst* argument is the address of an item list of descriptors used to describe the information to be formatted, as well as how the formatting is to be done.

For each item code, include an item descriptor and terminate the list with a longword containing the value of 0. For a description of *item_list_3,* please refer to the section on data types in the *Introduction to VMS System Routines* manual.

The following item codes are used with $FORMAT_CLASS. See Table 3-1 for information on the format of items.

Item Code	Use
CLS$_CLSBLK	The classification block to be formatted.
CLS$_MINCLSBLK	The minimum classification block to be formatted. (It has the same format as CLS$_CLSBLK.)
CLS$_MAXCLSBLK	The maximum classification block to be formatted. (It has the same format as CLS$_CLSBLK.)
CLS$_WIDTH	The maximum width of the display. If this is zero or omitted, the output strings will be long strings with no line breaks.

Item Code	Use
CLS$_TRMDSC	A string of characters used to terminate each line (as determined by CLS$_WIDTH) of the output strings. This is normally a <CR> <LF> sequence. If this is null or omitted, the output strings will be long strings with no line breaks.
CLS$_INDENT	The number of spaces to prefix each line (as determined by CLS$_WIDTH) of the output strings. This indents the output strings.
CLS$_SECSTR	A text string representing the secrecy class.
CLS$_INTSTR	A text string representing the integrity class.

Note: **When a line width is specified and exceeded, multiple lines are created in the output buffer by appending the CLS$_TRMDSC string to the end of each line segment. The beginning of each line segment is then indented the number of spaces specified by CLS$_INDENT.**

RETURN VALUES

SS$_NORMAL	The service completed successfully.
SS$_ACCVIO	One of the input buffers or its descriptor cannot be read, the output descriptors cannot be read, the output buffers cannot be written, or one of the output length words cannot be written.
SS$_BUFFEROVF	The service successfully completed. One of the output buffers has overflowed and been truncated. This is indicated by a -1 as the length of the return string.

$CHANGE_CLASS

The CHANGE CLASS service allows the suitably privileged user to observe or modify the classification of any object.

FORMAT

SYS$CHANGE_CLASS *objtyp, objnam, itmlst, [acmode]*

RETURNS

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

Longword condition value. All system services return (by immediate value) a condition value in R0. Condition values that can be returned by this service are listed under "Return Values".

ARGUMENTS

objtyp

Address of a longword containing an object type code. These are the same system objects supported by the $CHANGE_ACL system service and the same object type codes are used. See the *VMS System Services Volumes* for more information. The symbols are defined in the system marco library ($ACLDEF). The values and meanings of these symbols are defined in the following table.

The following type codes are valid:

ACL$C_FILE	The object is an ODS-2 file.
ACL$C_DEVICE	The object is any device with an associated UCB.
ACL$C_LOGICAL_NAME_TABLE	The object is a logical name table.
ACL$C_SYSTEM_GLOBAL_SECTION	The object is a system global section.
ACL$C_GROUP_GLOBAL_SECTION	The object is a group global section.
ACL$C_JOBCTL_QUEUE	The object is a queue.
ACL$C_PROCESS	The object is a process. [1]

[1] A process is a subject, not an object. This code is provided for ease of use only.

objnam

Address of a string descriptor pointing to the name of the object. For a process, the *objnam* argument can be the address of an unsigned longword descriptor (DSC$K_DTYPE_LU) pointing to a process ID.

itmlst

Address of a list of descriptors specifying the object's new access class (minimum and maximum) and pointers to where the object's original access class is to be saved.

The item list is used to specify the new access class for the specified object and buffers in which the previous access class may be placed.

The symbols are defined in the system marco library ($CLSDEF). The values and meanings of these symbols are defined in the following table.

Item Code	Use
CLS$_CLSBLK[1]	The binary classification block specifying the new classification.
CLS$_MINCLSBLK[1]	The binary classification block specifying the new minimum classification. (It has the same format as CLS$_CLSBLK.)
CLS$_ MAXCLSBLK[1]	The binary classification block specifying the new maximum classification. (It has the same format as CLS$_CLSBLK.)
CLS$_OLDCLSBLK	A classification block to contain the old classification. (It has the same format as CLS$_CLSBLK.)
CLS$_OLDMINCLS	A classification block to contain the old minimum classification. (It has the same format as CLS$_CLSBLK.)
CLS$_OLDMAXCLS	A classification block to contain the old maximum classification. (It has the same format as CLS$_CLSBLK.)

[1] If *objtyp* is specified as ACL$L_PROCESS, an SS$_BADPARAM error occurs.

acmode

Address of a byte containing the access mode in which the arguments are validated. This is maximized with the mode of the caller.

DESCRIPTION

The $CHANGE_CLASS system service is used to observe or modify the classification associated with any defined object. However, for this service to succeed, the process must have the appropriate privilege (UPGRADE, DOWNGRADE, or BYPASS). Process classifications cannot be modified.

In regard to this, note the following information:

1 No new classification blocks need be specified; the service then acts as "GET_CLASS".

2 The protection checking is done as follows:

   - The process must have read access to an object to obtain or change its classification. If just the old classification is desired, this is the only check.
   - The process must have read and control access to change the classification.
   - The process must have the SECURITY privilege to set a classification range. SEVMS requires SECURITY to further restrict the ability to create a persistent channel.
     - Without SECURITY:
       - If a new maximum classification only is specified, it must equal the old minimum.

    - If a new minimum only is specified, it must equal the old maximum.
    - If both an new minimum and maximum are specified, they must be equal.
    - If the object has no valid classification (the ORB$V_CLASS_PROT flag is off) then the new minimum must equal the new maximum. If only one is specified, the other is set equal.
    - If any of these checks fail, SS$_NOSECURITY is returned.
- With SECURITY:
    - If a new maximum classification only is specified, it must dominate the old minimum.
    - If a new minimum only is specified, it must be dominated by the old maximum.
    - If the object has no valid classification and a new minimum or maximum only is specified, the other is set equal.
    - If both a new minimum and maximum are specified, the maximum must dominate the minimum.
    - If any of the checks above fail, SS$_BADPARAM is returned.
    - If the minimum secrecy is not equal to the maximum secrecy, DOWNGRADE is required; SS$_NODOWNGRADE is returned if the process does not have it.
    - If the minimum integrity is not equal to the maximum integrity, UPGRADE is required; SS$_NOUPGRADE is returned if the process does not have it.
- Unless the process has BYPASS or the object has no valid classification:
    - A process running at the old minimum classification, with the privileges of the current process, would have write access to the newly classified object. In other words, for a non-privileged process, the resultant minimum classification must dominate the object's old minimum classification; otherwise, DOWNGRADE (and/or UPGRADE) is required.
    - If the check fails, SS$_NODOWNGRADE (or SS$_NOUPGRADE) is returned.

RETURN VALUES

SS$_NORMAL	The service completed successfully.
9	The service completed successfully, but the object did not have a valid class block prior to service execution.

SS$_ACCVIO

The object type could not be read, the object name buffer or its descriptor could not be read, the classification block could not be read, the access mode byte could not be read, or the old classification block buffer or its descriptor could not be written.

SS$_NOUPGRADE

An attempt was made to upgrade the access class of an object without the necessary UPGRADE or BYPASS privilege.

SS$_NODOWNGRADE

An attempt was made to downgrade the access class of an object without the necessary DOWNGRADE or BYPASS privilege.

SS$_BUFFEROVF

The service completed successfully. However the buffer specified to receive the old classification block overflowed and has been truncated.

3.3 SEVMS Binary Audit Record Format

The binary audit record format used by SEVMS is the same as that used by VMS; it is described in full in "Appendix A - Security Audit Message Format" of the *VMS Audit Analysis Utility Manual.*

In addition to the record types and packets described in the *VMS Audit Analysis Utility Manual,* SEVMS adds several new record types and packets. Information about these record types and packets is provided in the following subsections of this section.

3.3.1 SEVMS Audit Record Types and Subtypes

For SEVMS-unique audit records, the NSA$W_FACILITY field in the audit header packet (described in *VMS Audit Analysis Utility Manual*) will contain the SEVMS facility code of 1062. When this field contains this SEVMS facility code, the record type and record subtype fields of the audit header packet will contain the SEVMS-unique information described in Table 3-2, which follows.

Table 3-2 SEVMS-unique Audit Record Types and Subtypes

Event Type	Event Class	Record Type	Record Subtype
Label bypass	success	1	1
Label bypass	fail	1	2
Printed file	success	1	4
Printed file	fail	1	3
Change class	success	2	1
Change class	fail	2	2

3.3.2 SEVMS Audit Data Packets

This section describes the audit data packets used by SEVMS.

The following table, Table 3-3, contains the values which are defined for SEVMS-unique packets. These packets are in addition to the packets described in "Section A.2 - Audit Data Packets" of the *VMS Audit Analysis Utility Manual.*

Table 3-3 SEVMS Audit Data Packets

Packet Type	Code	Meaning
OBJECT_MIN_CLASS	27	Minimum classification of an object. (Classification block)[6]
OBJECT_MAX_CLASS	28	Maximum classification of an object. (Classification block)[6]

[6] The format is described in "Appendix A" of the *VMS Audit Analysis Utility Manual.*

Table 3-3 (Cont.) SEVMS Audit Data Packets

Packet Type	Code	Meaning
NEW_MIN_CLASS	120	New minimum classification of an object (or attempted new minimum classification of an object). (Classification block)[6]
NEW_MAX_CLASS	121	New maximum classification of an object. (or attempted new maximum classification of an object). (Classification block)[6]
QUEMGR_JOB_NUMBER	123	Queue manager job number. (Longword)
SYMBIONT_PROCESS_ID	124	Symbiont process ID number. (Longword)

[6] The format is described in "Appendix A" of the *VMS Audit Analysis Utility Manual.*

A SEVMS System Messages and Recovery Procedures

This section contains a listing of the errors, warnings, and informational messages which are issued by SEVMS. Included are a brief description of each message and the action to be taken in response to the message.

These system messages are unique to SEVMS; VMS system messages are described in the *VMS System Messages and Recovery Procedures Reference Volume*, which is a part of the VMS General User Subkit.

The format of SEVMS system messages is the same as that used for VMS system messages. For format information about system messages, refer to the *VMS System Messages and Recovery Procedures Reference Volume* which is a part of the VMS General User Subkit.

In the following descriptions of SEVMS system messages, a **?** is used to represent the severity code which occurs in the message. In an actual system message, one of the following severity codes will appear in the message:

Severity Code	Meaning
E	error
F	fatal error
I	informational
S	success
W	warning

%SEVMS-?-BADFAOARG

Explanation: An error was encountered in an internal $FAO call.

User Action: Submit an SEVMS SPR.

%SEVMS-?-BADFLARG

Explanation: The SEVMS Print Symbiont was unable to access the routine and/or shareable image specified in a .FORMAT_LINE template directive.

User Action: Check that the routine and shareable image names were specified correctly. Check that the routine entry point is a universal symbol.

%SEVMS-?-BADSYNTAX

Explanation: Insufficient or illegal arguments were specified in a SET TEMPLATE command.

%SEVMS-?-BUFOVRFLO

Explanation: A line of a referenced template was greater than 512.

User Action: Reduce the length of the template line.

%SEVMS-?-CATOUTRNG

Explanation: Categories must be between 1 and 128. An out-of-range category number has been selected.

User Action: Use a valid category.

%SEVMS-?-CLASSPROTNEEDED

Explanation: The CLASS_PROT parameter must be on to perform this operation.

User Action: The dynamic SYSGEN parameter CLASS_PROT must be set to "1" to enable mandatory access controls.

%SEVMS-?-CLASSPROTOFF

Explanation: The CLASS_PROT parameter is off - classification will be ignored.

User Action: CLASS_PROT must be set to "1" to classify objects successfully.

%SEVMS-?-CLSHDRFRM

Explanation: An error was encountered closing SEVMS$SMB_HDRFRM.DAT. Additional error messages (such as RMS errors) should also be displayed.

User Action: Correct the problem indicated by the additional error information.

%SEVMS-?-DEVCLS

Explanation: An error was encountered in an internal $CHANGE_CLASS call.

User Action: Submit an SEVMS SPR.

%SEVMS-?-ERREADFRM

Explanation: An error was encountered accessing or parsing a record of a template in SEVMS$SMB_LIB.TLB.

%SEVMS-?-ERREADHDR

Explanation: An error was encountered reading SEVMS$SMB_HDRFRM.DAT. Additional error messages should be displayed.

User Action: Correct the problem indicated by the additional messages.

%SEVMS-?-ERRONQUE

Explanation: An error was encountered setting up a print queue. Additional error messages should be displayed.

User Action: Correct the problem indicated by the additional messages.

%SEVMS-?-ERRWRTHDR

Explanation: An error was encountered writing SEVMS$SMB_HDRFRM.DAT. Additional error messages should be displayed.

User Action: Correct the problem indicated by the additional messages.

%SEVMS-?-LEVOUTRNG

Explanation: Levels must be between 0 and 255. An out-of-range level number has been selected.

User Action: Use a valid level.

%SEVMS-?-LINTOOLNG

Explanation: A line of a referenced template was too long for the width of the printer.

User Action: Print the file on a wider printer or reformat the template so that shorter lines are generated.

%SEVMS-?-MININTGTRMAX

Explanation: The minimum integrity classification specified is greater than the maximum.

User Action: Specify aminimum integrity classification level that is no greater than the maximum.

%SEVMS-?-MINSECGTRMAX

Explanation: The minimum security classification specified is greater than the maximum.

User Action: Specify a minimum security classification level that is no greater than the maximum.

%SEVMS-?-NOCLASSSUPPORT

Explanation: Classification not supported on the selected object.

User Action: NONE. The selected object (for example, an event flag cluster) cannot be classified with SET CLASS.

%SEVMS-?-NOFORMCAT

Explanation: There is no template associated with the specified security category.

%SEVMS-?-NOLABELS

Explanation: A file is being printed with no page headers and/or page trailers, using the /PASSALL switch.

%SEVMS-?-NOMODULE

Explanation: The template associated with a specific security category was not found in SEVMS$SMB_LIB.TLB.

User Action: A template must be specified for the particular set of categories using the SET TEMPLATE command.

%SEVMS-?-NOPASSALL

Explanation: An attempt was made to print a file using the /PASSALL qualifier on a queue where the qualifier was not allowed.

User Action: See the *VMS SES Security Manager's Guide* for more information on the print symbiont.

%SEVMS-?-OPNHDRFRM

Explanation: An error was encountered opening SEVMS$SMB_HDRFRM.DAT. Additional error messages should be displayed.

User Action: Correct the problem indicated by the additional messages.

%SEVMS-?-OPNLIBRARY

Explanation: An error was encountered opening SEVMS$SMB_LIB.TLB. Additional error messages should be displayed.

User Action: Correct the problem indicated by the additional messages.

%SEVMS-?-PARSECLAS

Explanation: An error was encountered converting a class block to a text string.

User Action: Submit an SEVMS SPR.

%SEVMS-?-PRINTTERM

Explanation: A print job was terminated due to error. Additional error messages should be displayed.

User Action: Correct the problem indicated by the additional messages.

%SEVMS-?-SMALLHEADER

Explanation: The file header has no space allocated for classification.

User Action: Either:

- COPY the file to a new version (assuming CLASS_PROT is set to "1"), or
- image backup and restore the entire disk with the SEVMS version of the BACKUP utility.

%SEVMS-?-TOOMANVAR

Explanation: More .PAGE directives were specified in a template definition than is currently supported.

User Action: Reduce the number of .PAGE directives used in the template. Report this in an SPR so that the number may be increased in future versions of SEVMS.

%SEVMS-?-USRFLERR

Explanation: The customer-supplied formatting routine returned an error when called by a .FORMAT_LINE template directive.

User Action: Take action appropriate for the specific error returned by the customer-supplied formatting routine.

Glossary

This glossary contains definitions of terms used in VMS SES documentation. Included are terms which pertain to SEVMS, or which pertain to computer security in general.

To provide consistency and avoid confusion in VMS SES documentation, many of the definitions of terms included in this glossary have been taken from the *Department of Defense Trusted Computer System Evaluation Criteria,* DoD 5200.28-STD.

Some terms used in VMS SES documentation are described in VMS documentation, in the VMS General User Subkit, *General User Volume - Glossary,* and are not repeated in this glossary.

access: A specific type of interaction that takes place between a subject and an object that permits the flow of information from one to the other.

ACL: See *Access Control List.*

access control list: A list that defines the types of access to be granted or denied to users of an object - used for discretionary access control. Access control lists can be created for objects such as files, VMS devices, and mailboxes. Each access control list consists of one or more entries known as access control list entries.

access controls: See *mandatory access controls* and *discretionary access controls.*

access rights block (ARB): A VMS data structure associated with a process containing mandatory and discretionary access control information, as well as information about privileges.

alarm: See *security alarm.*

ARB: See *access rights block.*

auditing: See *security auditing.*

audit archive file: An SEVMS RMS sequential file with a binary record format in which audit records are stored for later analysis.

breakin attempt: An effort made by an unauthorized source to gain access to a system.

categories: See *secrecy categories.*

classification: The combination of secrecy and integrity levels and categories.

classification block (CLS$): The binary representation of a classification.

classification label: A piece of information that represents the security level of an object. It describes the classification (or sensitivity) of the data in the object. Classification labels are used by the Trusted Computing Base as the basis for mandatory access control decisions.

clearance: See *security clearance.*

data: Information with a specific physical representation.

discretionary controls: Security controls that are applied at the user's option; the system does not require their use. Access control lists (ACLs) are an example of such optional security features.

discretionary access control: A means of restricting access to an object based on the identity of subjects and/or groups to which they belong. The controls are discretionary in the sense that a subject with a certain access permission is capable of passing that permission (perhaps indirectly) on to any other subject (unless restrained by mandatory access control).

dominate: Security level S1 is said to dominate security level S2 if the hierarchical classification of S1 is greater than or equal to that of S2 and the non-hierarchical categories of S1 include all those of S2 as a subset.

integrity: Preservation of the trustworthiness of the data contained in a classified object. (i.e. An assurance that data has not been altered.)

integrity level: One of two factors which define the integrity of an object. SEVMS supports up to 255 integrity levels.

integrity category: One of two factors which define the integrity of an object. SEVMS supports up to 64 integrity categories.

label: See *classification label.*

level: See *integrity levels*. See also *secrecy levels.*

mandatory controls: See *mandatory access controls.*

mandatory access controls: A method of restricting access to an object based upon the sensitivity of the information contained in the object (determined by the object's classification label) and the authorization of a subject to access classified information (determined by the subject's classification).

multi-level device: A device that is used in a manner that permits it to simultaneously process data of two or more security levels without risk of compromise.

non-discretionary access controls: See *mandatory access controls.*

object: A system resource such as a file, device, or directory. A passive entity that contains or receives information. Access to an object implies access to the information it contains.

object rights block (ORB): A VMS data structure associated with an object containing access control information.

ORB: See *object rights block.*

read: A fundamental operation that results only in the flow of information from an object to a subject.

read-down: The subject can read an object with a lower classification than itself.

read-up: The subject can read an object with a higher classification than itself.

read access: Permission to read information.

secrecy: Preservation of the confidentiality of the data contained in a classified object.

secrecy category: One of two factors which define the secrecy of an object. SEVMS can have up to 128 secrecy categories. Secrecy categories are non-hierarchical.

secrecy level: One of two factors which define the secrecy of an object. SEVMS can have from 0 to 255 secrecy levels. Secrecy levels are hierarchical.

security alarm: A message sent to specified operator terminals which are enabled to receive security alarms. Security alarms are triggered by the occurrence of events previously designated by the security or system manager.

security auditing: The monitoring and recording of specified events occurring on the system. Examples of events which can be monitored are login failures, privileged and unprivileged acess to system objects, changes to the user authorization file, etc.

breach: A break in system security that results in admittance of a person or program to an object.

Security Enhancement Service (SES): See *VMS Security Enhancement Service.*

security manager: The person responsible for the enforcement of security policies, procedures, and practices on a computer system. SEVMS security management tasks are sometimes performed by the system manager.

security policy: A set of rules and practices that define and regulate how an organization manages, protects, and distributes sensitive information.

sensitivity label: See *classification.*

SES: See *VMS Security Enhancement Service.*

SEVMS: Licensed software component of VMS Security Enhancement Service.

single-level device: A device that is used to process data of a single security level at any one time. In SEVMS, it is a device assigned a single classification rather than a classification range.

subject: An active entity, generally in the form of a person, process, or device that causes information to flow among objects, or changes the system state.

system manager: The person responsible for the policies, procedures, and daily operation of a computer system. VMS system management tasks are sometimes performed by more than one person and might also include responsibilities for cluster management and security management.

Trusted Computing Base (TCB): The totality of protection mechanisms within a computer system - including hardware, firmware, and software - the combination of which is responsible for enforcing a security policy. A TCB consists of one or more components that together enforce a unified security policy over a product or system. The ability of a TCB to correctly enforce a security policy depends solely upon the mechanisms within the TCB and on the correct input by system administrative personnel of paramenter (such as a user's clearance) related to the security policy.

VMS Security Enhancement Service (VMS SES): A software security consulting product. Provides a means of devising a system-wide security policy using mandatory access controls. Combines consulting services with packaged application software and documentation.

VMS SES: See *VMS Security Enhancement Service*

write-down: The subject can write to an object with a lower classification than itself.

write-up: The subject can write to an object with a higher classification than itself.

write access: Permission to write to an object.

Index