

# AllFusion™ Endeavor® Change Manager

---

Security Guide  
4.0



Computer Associates™

This documentation and related computer software program (hereinafter referred to as the "Documentation") is for the end user's informational purposes only and is subject to change or withdrawal by Computer Associates International, Inc. ("CA") at any time.

This documentation may not be copied, transferred, reproduced, disclosed or duplicated, in whole or in part, without the prior written consent of CA. This documentation is proprietary information of CA and protected by the copyright laws of the United States and international treaties.

Notwithstanding the foregoing, licensed users may print a reasonable number of copies of this documentation for their own internal use, provided that all CA copyright notices and legends are affixed to each reproduced copy. Only authorized employees, consultants, or agents of the user who are bound by the confidentiality provisions of the license for the software are permitted to have access to such copies.

This right to print copies is limited to the period during which the license for the product remains in full force and effect. Should the license terminate for any reason, it shall be the user's responsibility to return to CA the reproduced copies or to certify to CA that same have been destroyed.

To the extent permitted by applicable law, CA provides this documentation "as is" without warranty of any kind, including without limitation, any implied warranties of merchantability, fitness for a particular purpose or noninfringement. In no event will CA be liable to the end user or any third party for any loss or damage, direct or indirect, from the use of this documentation, including without limitation, lost profits, business interruption, goodwill, or lost data, even if CA is expressly advised of such loss or damage.

The use of any product referenced in this documentation and this documentation is governed by the end user's applicable license agreement.

The manufacturer of this documentation is Computer Associates International, Inc.

Provided with "Restricted Rights" as set forth in 48 C.F.R. Section 12.212, 48 C.F.R. Sections 52.227-19(c)(1) and (2) or DFARS Section 252.227-7013(c)(1)(ii) or applicable successor provisions.

**First Edition, December 2002**

©2002 Computer Associates International, Inc.  
All rights reserved.

All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

# Contents

---

<b>Chapter 1. Security Overview</b>	1-1
1.1 What Are Your Security Options?	1-2
1.1.1 Data Set Security	1-2
1.1.2 Functional Security	1-3
1.1.2.1 Security Control Points	1-3
1.1.2.2 Native Security Tables	1-4
1.1.2.3 Endeavor External Security Interface (ESI)	1-5
1.2 Selecting a Security Option for Your Installation	1-7
1.2.1 Data Set Security Methods	1-7
1.2.2 Functional Security Methods	1-7
1.3 How to Implement Security	1-9
1.3.1 Enabling Data Set Security	1-9
1.3.2 Enabling Functional Security	1-9
1.3.2.1 Native Security	1-9
1.3.2.2 ESI Security	1-10
1.4 Documentation Overview	1-11
1.5 Syntax Conventions	1-12
1.5.1 Sample Syntax Diagram	1-14
1.5.2 Syntax Diagram Explanation	1-15
1.5.3 General Coding Information	1-17
1.5.3.1 Valid Characters	1-17
1.5.3.2 Incompatible Commands and Clauses	1-18
1.5.3.3 Ending A Statement	1-18
1.5.3.4 SCL Parsing Information	1-18
<b>Chapter 2. Implementing Data Set Security</b>	2-1
2.1 Data Set Security	2-2
2.1.1 Data Set Security Using Alternate ID Support	2-2
2.1.1.1 Data Sets Controlled by the Alternate User ID	2-2
2.1.1.2 Data Sets Not Controlled by the Alternate User ID	2-3
2.2 Activating Data Set Protection	2-4
2.3 Program Pathing Using eTrust CA-ACF2 and eTrust CA-Top Secret	2-5
<b>Chapter 3. Enabling Endeavor Native Security</b>	3-1
3.1 Native Security Tables	3-2
3.2 Security Control Points	3-3
3.2.1 Environment Selection Security Control Point	3-5
3.2.2 Primary Options Security Control Point	3-5
3.2.3 Foreground Options Security Control Point	3-5

3.2.4	Action Initiation Security Control Point	3-6
3.2.5	Package Actions Security Control Point	3-6
3.3	User Exit Modules	3-7
3.4	How Endeavor Reads the Security Tables	3-8
3.4.1	User Access	3-8
3.4.2	System or Subsystem Access	3-8
3.4.3	Resource Restrictions	3-8
3.5	Implementing Native Security: A Quick Reference	3-9
3.5.1	Defining Endeavor Native Security Tables	3-9
3.5.2	Entering CONSDEF Macros	3-9
3.5.2.1	Coding Conventions	3-10
3.5.2.2	Order of Security Definitions	3-10
3.6	Defining the Access Security Table	3-11
3.6.1	Overview	3-11
3.6.1.1	type=start,table=user	3-12
3.6.1.2	type=end,table=user	3-12
3.6.1.3	type=user	3-12
3.6.2	type=user Parameters	3-12
3.6.2.1	userid=userid	3-12
3.6.2.2	group=group-name	3-12
3.6.2.3	sysdef=((stage1-name,\$,r))	3-12
3.6.3	Assembling and Link-Editing the Access Security Table	3-13
3.6.4	Update the Defaults Table	3-13
3.7	Defining the User Security Table	3-15
3.7.1.1	type=start,table=user	3-16
3.7.1.2	type=end,table=user	3-16
3.7.1.3	type=user	3-16
3.7.2	type=user Parameters	3-16
3.7.2.1	userid=user-id	3-16
3.7.2.2	group=group-name	3-16
3.7.2.3	[until=yyddd]	3-17
3.7.2.4	sysdef=( (sys-name,subsys-name,access1,access2)...)	3-17
3.7.3	Assembling and Link-Editing the User Security Table	3-20
3.8	Defining the Resource Security Table	3-22
3.8.1.1	type=start,table=resource	3-23
3.8.1.2	type=end,table=resource	3-23
3.8.1.3	type=resource	3-23
3.8.2	type=user Parameters	3-23
3.8.2.1	rname=element-name	3-23
3.8.2.2	group=group-name	3-23
3.8.2.3	[until=yyddd]	3-24
3.8.2.4	sysdef=( (sys-name,subsys-name,access1,access2)...)	3-24
3.8.3	Assembling and Link-Editing the Resource Security Table	3-25
3.9	Modifying the Defaults Table	3-27
3.9.1	Procedure	3-27
<b>Chapter 4.</b>	<b>Endeavor External Security Interface (ESI)</b>	<b>4-1</b>
4.1	Endeavor ESI Overview	4-2
4.1.1	Security Checkpoints	4-2
4.1.2	How ESI Security Works	4-2
4.1.2.1	ESI Defaults Entries (ESIDFLTS)	4-3

4.1.2.2	Function Equates Entries (FUNCEQU)	4-3
4.1.2.3	Name Equates Entries (NAMEQU)	4-3
4.1.2.4	The Security Processing Model	4-4
4.1.3	User Exit Modules	4-5
4.2	The Name Equates Table	4-6
4.2.1	Defining ESI Diagnostics	4-8
4.2.2	Defining SAF Authorization Levels	4-9
4.2.2.1	Mapping Authorization Values to RACF, eTrust CA-ACF2, and eTrust CA-Top Secret	4-11
4.2.2.2	FUNCEQU Authorization	4-12
4.2.3	Defining SAF Name Formats	4-13
4.2.3.1	Changing Names Generated at Security Control Points	4-14
4.2.3.2	Specifying a Substring of a Keyword	4-16
4.2.3.3	Defining a Class Other Than Data Set with RACF	4-21
4.2.4	Security Rule Formats ENVIRONMENT_ACCESS Through PACKAGE_ACTIONS	4-23
4.2.4.1	ENVIRONMENT_ACCESS—Environment Access Security Control Point	4-25
Example		4-25
4.2.4.2	PRIMARY_OPTIONS — Primary Options Security Control Point	4-26
Example		4-27
4.2.4.3	ACTION_INITIATION — Action Initiation Security Control Point (Standard)	4-27
4.2.4.4	ACTION_INITIATION — Action Initiation Security Control Point (Extension)	4-28
Example		4-28
4.2.4.5	The Default Authorization Value	4-28
4.2.4.6	PACKAGE_ACTIONS — Package Actions Security Control Point	4-30
4.2.4.7	PACKAGE_ACTIONS ESI Calls	4-32
4.3	Enabling Endeavor ESI	4-34
4.3.1	Preparing Your ESI Security Worksheet	4-34
4.4	Defining Security Rules for Your Site	4-35
4.4.1	Defining Security Rules for Your Site Environments	4-35
4.4.1.1	User Abbreviations	4-36
4.4.2	Defining Security Rules for the Primary Options Panel	4-37
4.4.3	Defining Security Rules for Action Initiations	4-38
4.4.4	Defining Security Rules for Your Package Actions Panel	4-40
4.4.5	Defining Rules for Your Site Security Package	4-41
4.4.5.1	What Happens if Site Security Rules Are Not Defined	4-41
4.4.5.2	Guidelines for Writing Security Rules	4-41
4.4.6	What to Do After You Fill Out Your ESI Worksheet	4-42
4.4.6.1	Developing ESI Profiles	4-42
4.4.6.2	Identifying User Ids That Require Access	4-43
4.4.6.3	Customizing Your NAMEQU Entries	4-43
4.4.7	Assemble and Link the Name Equates Table	4-43
4.4.7.1	Defining Rules for Your Site Security Package	4-43
4.4.7.2	Modifying the BC1JNEQU Member	4-44
4.4.7.3	Assembling and Linking the Modified Table	4-44
4.4.7.4	Refreshing the LINKLIST	4-44
4.4.8	Activating ESI Using the Endeavor C1DEFLTS Table	4-47

4.4.8.1	The ACCSTBL Field	4-47
4.4.8.2	The ESSI Field	4-47
4.4.8.3	The PKGSEC Field	4-47
4.4.8.4	Package Security, External Approvers, and Approver Groups	4-47
4.4.8.4	Modifying the Name Equates Table Name	4-48
4.5	Testing ESI Security and Monitoring Warnings	4-50
4.5.1	How to Verify ESI Activation	4-50
4.5.2	Monitoring Security Violations	4-51
4.6	Using Endeavor ESI Warning Mode	4-52
4.6.1	ESI Defaults (ESIDFLTS) Macro	4-52
4.6.2	Enabling ESI Warning Mode	4-53
4.7	The Endeavor ESI Trace Facility	4-54
4.7.1	Ensuring the Correct Format of the Pseudo Data Set Name	4-54
4.7.2	Reviewing RACROUTE Request Return Codes	4-55
4.7.3	The Trace Record Format	4-55
4.7.4	Using the Trace Facility	4-56
4.7.4.1	Accessing the CLIST Library	4-56
4.7.4.2	ESITRACE Command in Foreground Mode	4-56
4.7.4.3	Running ESI Trace	4-58
4.7.4.4	Deactivating the Trace Facility	4-58
4.7.4.5	TSO ALLOCATE Command Tasks	4-58
4.7.4.6	ESI Trace in Batch Mode	4-59
<b>Appendix A. Security Worksheets</b>		A-1
A.1	Environment Security Control Worksheet	A-2
A.1.1	Part 1	A-2
A.1.2	Part 2	A-2
A.1.3	Part 3	A-2
A.1.4	Part 4	A-2
A.2	Primary Options Security Control Worksheet	A-3
A.2.1	Part 1	A-3
A.2.2	Part 2	A-3
A.2.3	Part 3	A-3
A.2.4	Part 4	A-3
A.3	Foreground Options Security Control Worksheet	A-4
A.3.1	Part 1	A-4
A.3.2	Part 2	A-4
A.3.3	Part 3	A-4
A.3.4	Part 4	A-4
A.4	Action Initiation Security Control Worksheet	A-5
A.4.1	Part 1	A-5
A.4.2	Part 2	A-5
A.4.3	Part 3	A-5
A.4.4	Part 4	A-5
A.5	Package Actions Security Control Worksheet	A-6
A.5.1	Part 1	A-6
A.5.2	Part 2	A-8
A.5.3	Part 4	A-8
<b>Appendix B. Endeavor ESI Logic Flow</b>		B-1
B.1	Endeavor ESI Logic Flow Diagram	B-2

**Index** ..... X-1



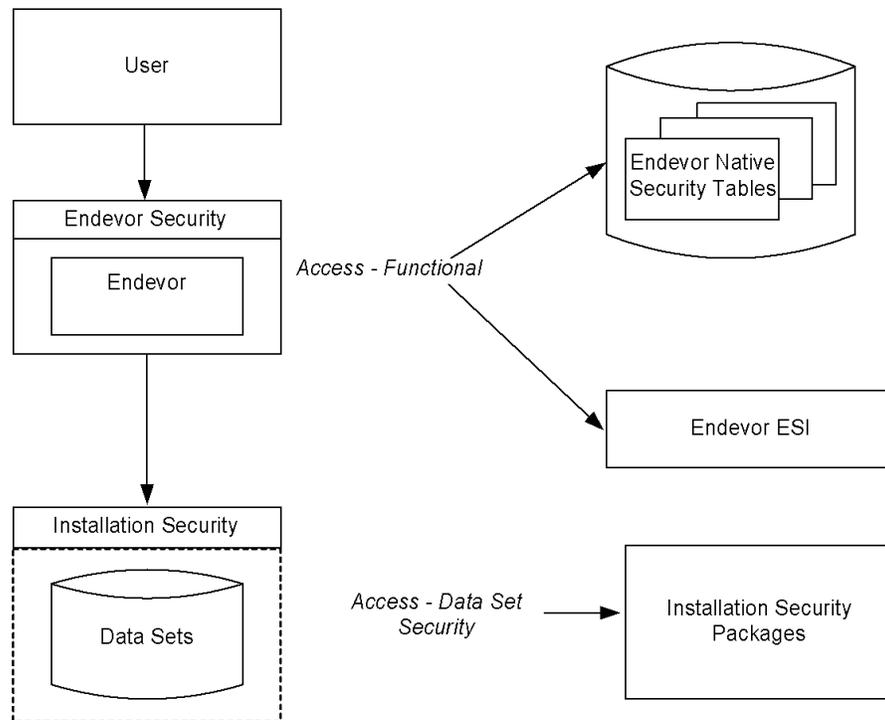
# Chapter 1. Security Overview

---

## 1.1 What Are Your Security Options?

To provide a comprehensive security program for your Endeavor system, you must address security issues in two essential areas: data set security and functional security.

The following figure depicts the relationship between Endeavor and functional and data set security. Note that Endeavor uses one of two available means of providing functional security: native security tables or Endeavor ESI.



### 1.1.1 Data Set Security

**Endeavor does not provide data set security.** Data set security is performed by a site security package, such as:

- RACF
- eTrust CA-ACF2
- eTrust CA-Top Secret

Data set security involves preventing unauthorized access to the data sets that Endeavor uses.

Your site security package (eTrust CA-ACF2, RACF, or eTrust CA-Top Secret) provides data set security. Two approaches are available for controlling access to your physical data sets:

- **Program path protection** — Gives the Endeavor system access to the data sets it maintains. Users must go through Endeavor to perform maintenance on these data sets.
- **Standard data set security** — Gives users direct access to data sets maintained by Endeavor. Although unauthorized access to data sets is prevented, authorized users can maintain these data sets without going through Endeavor.

Implementing data set security in addition to Endeavor functional security is recommended to control access to data sets by Endeavor users. See *Implementing Data Set Security* for information on implementing data set security.

## 1.1.2 Functional Security

Functional security involves protecting Endeavor inventory functions from unauthorized access. These functions include access to menu options, the ability to perform certain actions against certain inventory areas, and other secured Endeavor options.

Functional security is provided by Endeavor. You must choose between one of two methods for providing functional security:

- **Endeavor Native Security Tables** — Control environment access, primary and foreground menu options, and action authorization.
- **Endeavor External Security Interface (ESI)** — Controls environment access, primary and foreground menu options, action authorization, and package actions, as well as allowing you to store security rules under your site security package. In addition, ESI allows you to customize your functional security capabilities.

### 1.1.2.1 Security Control Points

During Endeavor processing, Endeavor performs security checks to allow or deny a user access to certain inventory areas and inventory actions. These checkpoints are referred to as *security control points*.

The security control points listed below determine the appropriate level of access to system inventories and functions:

- **Environment Selection** — Verifies a user's access to a requested environment(s).
- **Primary Options** — Verifies a user's access to particular operations appearing in the Primary Options menu.
- **Foreground Options** — Verifies a user's access to actions available on the Foreground Options menu.
- **Action Initiation** — Verifies a user's access to actions such as DISPLAY, ADD/UPDATE, RETRIEVE, or GENERATE.
- **Package Actions** — Verifies a user's access to package actions such as CREATE, CAST, REVIEW, and EXECUTE (applies to Endeavor ESI only).

When security control points are reached, Endeavor checks access privileges defined in one of the following security configurations:

- Endeavor native security tables.
- Endeavor ESI, interfacing with an external security product such as RACF, *eTrust* CA-ACF2, or *eTrust* CA-Top Secret.

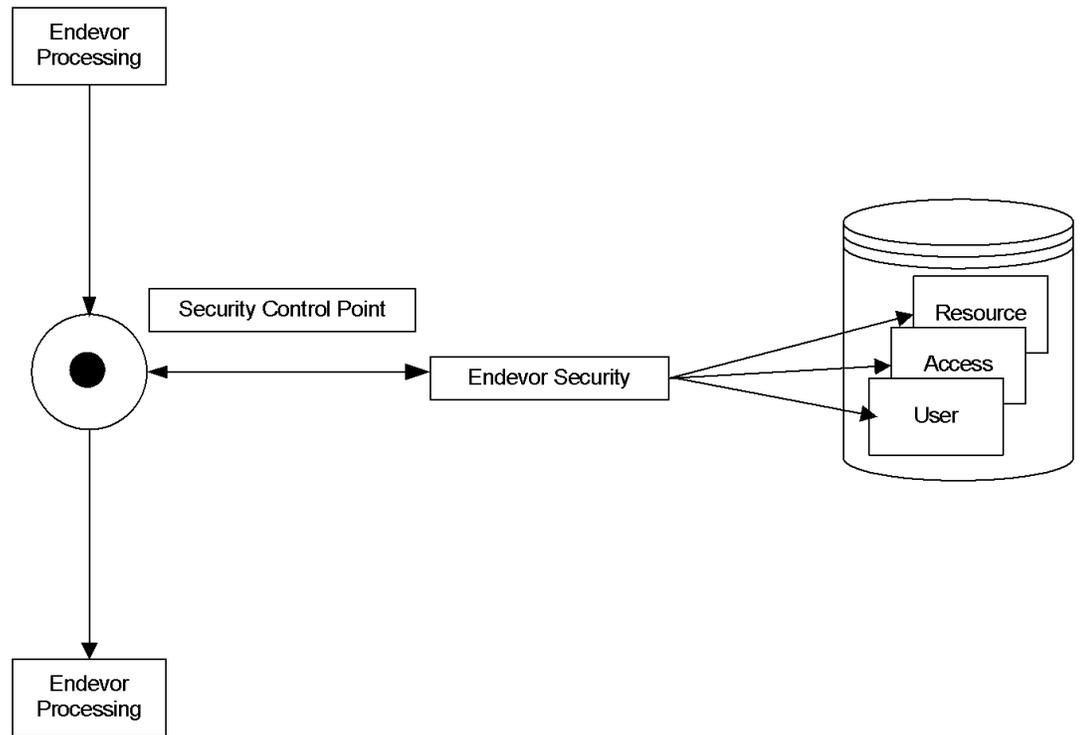
For example, when Endeavor reaches a security control point, the system reviews the native security tables and determines whether a user is allowed to perform certain inventory actions against a portion of the inventory. Endeavor then hides inventory elements the user is not permitted to access and functions the user is not allowed to perform. In this way, ESI allows you to customize your functional security capabilities.

### 1.1.2.2 Native Security Tables

Endeavor's native security uses the following three security tables to record security rules for access to inventory levels and functions:

- The **Access Security Table** (one table per installation) defines the environments to which a user has access.
- The **User Security Table** (one per environment) defines the menu options available to a user after access to an environment is obtained. Further, this table defines actions allowed within the environment, by user, for each system and subsystem.
- The **Resource Security Table** (one per environment) can be used to enforce naming conventions at the system/subsystem and element level.

The following figure shows how control points can control access to Endeavor processing functions by checking native security tables.

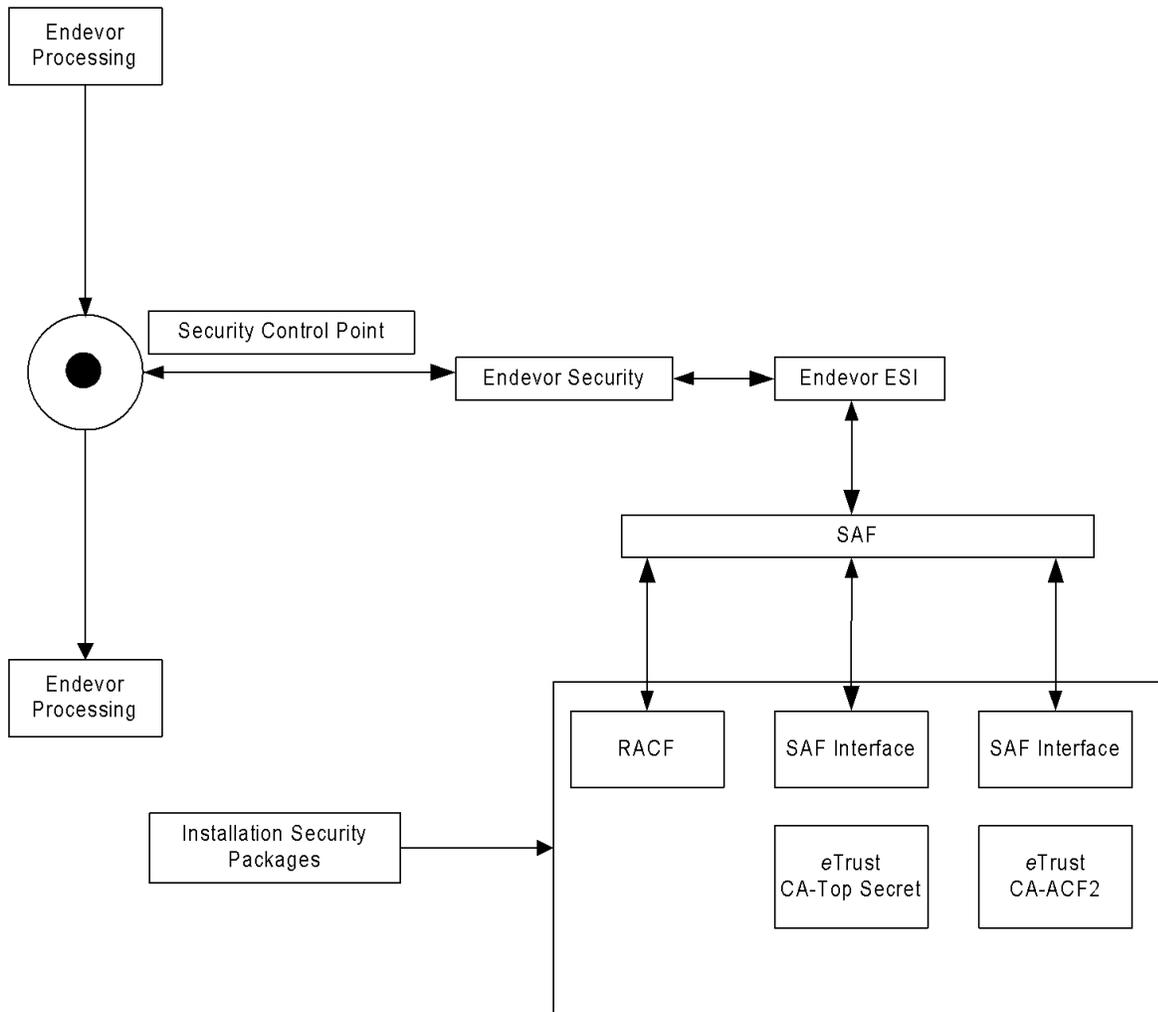


Upon entry into the system, a user's access to Endeavor functions is controlled by Endeavor Security Tables or the Endeavor External Security Interface (ESI). Site security provides protection of system data sets.

### 1.1.2.3 Endeavor External Security Interface (ESI)

Endevor ESI is an optional feature that allows you to secure Endeavor functions (access and actions) through IBM's System Authorization Facility (SAF), using the site security package on your system. It does this by allowing you to define the rules for functional security in your site security package (RACF, *eTrust* CA-ACF2, *eTrust* CA-Top Secret) rather than in the native tables supplied with Endeavor. Chapter 4, "Endevor External Security Interface (ESI)" explains and illustrates how to enable and use Endeavor ESI.

The following diagram shows how Endeavor ESI interacts with site security packages.



If Endevor ESI is enabled, security rules must be defined to the site security package (either RACF, *eTrust CA-ACF2*, or *eTrust CA-Top Secret*). In the previous diagram, Endevor uses IBM's System Authorization Facility (SAF) calls to query the installed security package instead of using Endevor native security tables.

## 1.2 Selecting a Security Option for Your Installation

To protect Endeavor and Endeavor data sets, you need to install a security function that prevents unauthorized access to your system. You may need to choose from among a number of data set and functional security options.

### 1.2.1 Data Set Security Methods

Options for data set security include the following:

#### **Program path protection**

- Allows authorized users access to a data set through an authorized program.
- Permits library updates only through Endeavor.
- Provides tight preventative control.

#### **Standard data set protection**

- Allows authorized users direct access to data sets.
- Permits library updates outside of Endeavor.
- Provides a measure of preventative control.

#### **No data set protection**

- Permits unlimited access to data sets.
- You can use Endeavor Footprint Exception Reporting to detect unauthorized updates. Refer to the *Reports Guide* for more information on reporting.

### 1.2.2 Functional Security Methods

Functional security can be provided by implementing one of the two following Endeavor security methods.

#### **Native security tables**

- Provide basic functional security.
- Allow access to environments.
- Permit primary and foreground menu options.
- Require authorization of actions.

#### **Endeavor ESI**

- Provides basic functional security.
- Allows extension of functional security through customization.

- Integrates with existing security package, such as RACF, *eTrust CA-ACF2*, or *eTrust CA-Top Secret*.

## 1.3 How to Implement Security

Computer Associates recommends a comprehensive approach to system security that ensures functional and data set security. A carefully planned security program ensures the proper levels of access and data set security.

You should address security during the final testing of your first Endeavor application. The initial setup and testing steps for implementing a new application should not be disrupted by overly restrictive security rules.

### 1.3.1 Enabling Data Set Security

To implement data set access security, the following steps are recommended:

1. Lay out your data set access requirements in a simple, non-technical
2. Build your data set security profiles in Warning Mode.
3. Test your security implementation and monitor any warnings you
4. Set data set security profiles to Live Mode.
5. Monitor security violations on an ongoing basis.

### 1.3.2 Enabling Functional Security

Decide which means of functional security you want to use: native security tables or Endeavor ESI. You can only use one method.

#### 1.3.2.1 Native Security

To implement Endeavor functional security using native security tables, the following steps are advised:

1. Plan security for a pilot application.
2. Define the three native security tables:
3. Access Security Table.
4. User Security Table.
5. Resource Security Table.
6. Activate the security tables.
7. Test your security implementation, monitor violations, and correct tables.

### **1.3.2.2 ESI Security**

To implement Endeavor ESI, the following steps are advised:

1. Plan security for a pilot application.
2. Customize the ESI Security Definition Table, BC1TNEQU.
3. Lay out ESI security profiles.
4. Build ESI security profiles in Warning Mode.
5. Customize the C1DEFLTS Table to activate ESI.
6. Test ESI security and monitor warnings.
7. Set ESI security profiles to Live Mode and monitor violations.

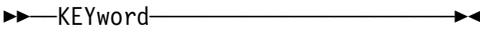
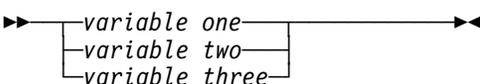
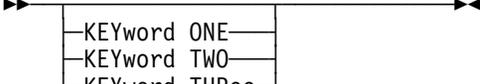
## 1.4 Documentation Overview

This manual is part of a comprehensive documentation set that fully describes the features and functions of Endeavor and explains how to perform everyday tasks. For a complete list of Endeavor manuals, see the PDF Table of Contents file in the PDF directory, or the Bookmanager Bookshelf file in the Books directory.

The following section describes product conventions.

## 1.5 Syntax Conventions

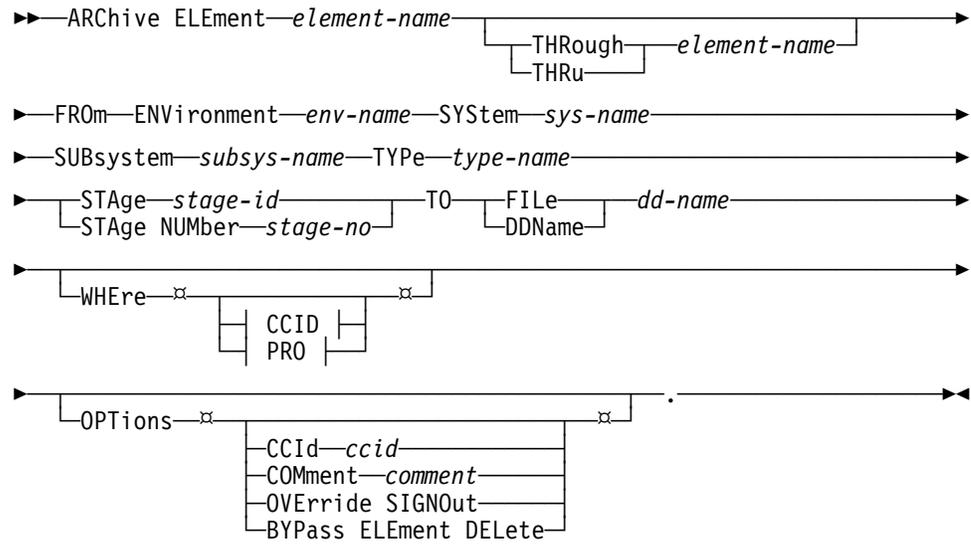
Endevor uses the IBM standard for representing syntax. The following table explains the syntax conventions:

Syntax	Explanation
	Represents the beginning of a syntax statement.
	Represents the end of a syntax statement.
	Represents the continuation of a syntax statement to the following line.
	Represents the continuation of a syntax statement from the preceding line.
	Represents a required keyword. Only the uppercase letters are necessary.
	Represents a required user-defined variable.
	Represents an optional keyword. Optional keywords appear below the syntax line. If coded, only the uppercase letters are necessary.
	Represents an optional user-defined variable. Optional variables appear below the syntax line.
	Represents a choice of required, mutually exclusive keywords. You must choose one and only one keyword.
	Represents a choice of required, mutually exclusive, user-defined variables. You must choose one and only one variable.
	Represents a choice of optional, mutually exclusive keywords. Optional keywords appear below the syntax line.

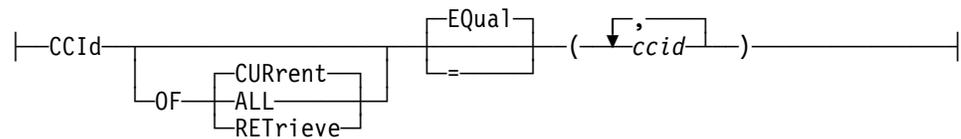
Syntax	Explanation
	Represents a choice of optional, mutually exclusive, user-defined variables. Optional variables appear below the syntax line.
	Represents a choice of optional keywords. The stars (x) indicate that the keywords are not mutually exclusive. Code no keyword more than once.
	Represents a choice of optional user-defined variables. The stars (x) indicate that the variables are not mutually exclusive. Code no variable more than once.
	Represents a choice of required, mutually exclusive keywords, one of which is the default. In this example, KEYword ONE is the default keyword because it appears above the syntax line.
	Represents a choice of required, mutually exclusive, user-defined variables, one of which is the default. In this example, <i>variable one</i> is the default variable because it appears above the syntax line.
	Represents a choice of optional, mutually exclusive keywords, one of which is the default. In this example, KEYword ONE is the default keyword because it appears above the syntax line.
	Represents a choice of optional, mutually exclusive, user-defined variables, one of which is the default. In this example, <i>variable one</i> is the default variable because it appears above the syntax line.
	Represents a required variable that can be repeated. Separate each occurrence with a comma and enclose any and all variables in a single set of parenthesis.

Syntax	Explanation
	Represents an optional variable that can be repeated. Separate each occurrence with a comma and enclose any and all variables in a single set of parenthesis.
	Represents a variable which must be enclosed by parenthesis.
	Represents a variable which must be enclosed by single quotes.
	Represents a variable which must be enclosed by double quotes.
	Represents a reference to a syntax fragment. Fragments are listed on the lines immediately following the required period at the end of each syntax statement.
<b>FRAGMENT:</b> 	Represents a syntax fragment.
	Represents the period required at the end of all syntax statements.

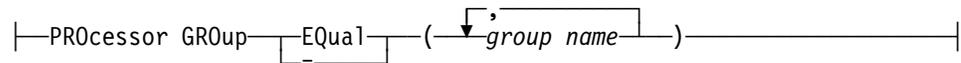
### 1.5.1 Sample Syntax Diagram



**CCID:**



**PRO:**



### 1.5.2 Syntax Diagram Explanation

Syntax	Explanation
ARChive ELEment <i>element-name</i>	The keyword ARChive ELEment appears on the main line, indicating that it is required. The variable <i>element-name</i> , also on the main line, must be coded.
THROUGH / THRu <i>element-name</i>	The keywords THROUGH and THRu appear below the main line, indicating that they are optional. They are also mutually exclusive.
FRom ENVironment ... TYPE <i>type-name</i>	Each keyword and variable in this segment appear on the main line, indicating that they are required.
STAge <i>stage-id</i> / STAge NUMber <i>stage-no</i>	The keywords STAge and STAge NUMber appear on and below the main line, indicating that they are required, mutually exclusive keywords.

Syntax	Explanation
TO ... <i>dd-name</i>	The keyword TO appears on the main line, indicating that it is required. The keywords FILE and DDName appear on and below the main line, indicating that they are required, mutually exclusive keywords. The variable <i>dd-name</i> also appears on the main line, indicating that it is required.
WHERE clause	This clause appears below the main line, indicating that it is optional. The keyword WHERE appears on the main line of the clause, indicating that it is required. CCID and PRO are syntax fragments that appear below the main line, indicating that they are optional. The stars (*) indicate that they are not mutually exclusive. For details on the CCID and PRO fragments, see the bottom of this table.
OPTion clause	This clause appears below the main line, indicating that it is optional. The keyword OPTion appears on the main line of the clause, indicating that it is required. The keywords CCId, COMment, OVErride SIGNOut, and BYPass ELEment DELEte all appear below the main line, indicating that they are optional. The stars (*) indicate that they are not mutually exclusive.
CCID fragment	<p>The keyword CCId appears on the main line, indicating that it is required. The OF clause appears below the main line, indicating that it is optional. If you code this clause, you must code the keyword OF, as it appears on the main line of the clause. CURrent, ALL, and RETrieve appear above, on, and below the main line of the clause, indicating that they are required, mutually exclusive keywords. CURrent appears above the main line, indicating that it is the default. If you code the keyword OF, you must choose one and only one of the keywords.</p> <p>The keywords EQual and = appear above and below the main line, indicating that they are optional, mutually exclusive keywords. EQual appears above the main line, indicating that it is the default. You can include only one. The variable <i>ccid</i> appears on the main line, indicating that it is required. The arrow indicates that you can repeat this variable, separating each instance with a comma. Enclose any and all variables in a single set of parenthesis.</p>

---

Syntax	Explanation
PRO fragment	The keyword PROcessor GROup appears on the main line, indicating that it is required. The keywords EQual and = appear on and below the main line, indicating that they are required, mutually exclusive keywords. You must include one. The variable <i>group name</i> appears on the main line, indicating that it is required. The arrow indicates that you can repeat this variable, separating each instance with a comma. Enclose any and all variables in a single set of parenthesis.

---

### 1.5.3 General Coding Information

In coding syntax, you must adhere to certain rules and guidelines regarding valid characters, incompatible commands and clauses, and ending statements. In addition, knowing how the SCL parser processes syntax helps you resolve errors and undesired results. The following sections outline these rules and guidelines.

#### 1.5.3.1 Valid Characters

The following characters are allowed when coding syntax:

- Uppercase letters
- Lowercase letters
- Numbers
- National characters
- Hyphen (-)
- Underscore (\_)

The following characters are allowed when coding syntax, but must be enclosed in either single (') or double (") quotation marks:

- Space
- Tab
- New line
- Carriage return
- Comma (,)
- Period (.)
- Equal sign (=)
- Greater than sign (>)
- Less than sign (<)

- Parenthesis ( )
- Single quotation marks
- Double quotation marks

A string containing single quotation marks must be enclosed in double quotation marks. A string containing double quotation marks must be enclosed in single quotation marks.

To remove information from an existing field in the database, enclose a blank space in single or double quotation marks. For example, the following statement removes the default CCID for user TCS:

```
DEFINE USER TCS  
DEFAULT CCID " ".
```

The characters "\*" and "%" are reserved for name masking.

### 1.5.3.2 Incompatible Commands and Clauses

The following commands and clauses are mutually exclusive:

- THROUGH and MEMBER clauses within any action except LIST
- Endeavor location information (environment, system, subsystem, type, and stage) and data set names (DSName)
- File names (DDName) and data set names (DSName)
- The stage id (STAge / STAge ID) and the stage number (STAge NUMBER)
- The SET TO Endeavor location information and the SET TO MEMBER clause

### 1.5.3.3 Ending A Statement

You must enter a period at the end of each statement. If no period is found, you receive an error message and the job terminates.

### 1.5.3.4 SCL Parsing Information

- The SCL parser does not look for information in columns 73-80 of the input. Therefore, be sure that all relevant information is coded in columns 1-72.
- The SCL parser does not catch duplicate clauses coded for an SCL request. If you code the same clause twice, SCL uses the Boolean "AND" to combine the clauses. If the result is invalid, you receive an error message.
- If you enter an asterisk (\*) in column 1, the remainder of the line is considered a comment by the SCL parser and is ignored during processing.
- Any value found to the right of the period terminating the SCL statement is considered a comment by the SCL parser and is ignored during processing.

## **Chapter 2. Implementing Data Set Security**

---

## 2.1 Data Set Security

Data set security refers to the protection of files accessed and maintained by Endeavor. Endeavor's Alternate ID feature allows you to restrict access to specified data sets by the user ID specified in the customer default table. It automatically determines which data sets are controlled by Endeavor. This practice allows Endeavor to perform updates only on the set of libraries you specify.

### 2.1.1 Data Set Security Using Alternate ID Support

At this time, some versions of program pathing do not support the ability to recognize top level programs through conventional means. A special interface has been developed in lieu of *eTrust CA-ACF2*, *eTrust CA-Top Secret*, and *RACF* to enable you to perform data set security using alternate ID support. This interface allows manipulation of data sets and their contents through Endeavor, using an alternate user ID.

This method of data set security ensures that updates to controlled data sets are performed only through Endeavor. External access to these files by Endeavor users is prohibited. **Computer Associates recommends using alternate ID support for data set security.**

#### 2.1.1.1 Data Sets Controlled by the Alternate User ID

When activated, the alternate Endeavor user ID is used when any of the following data set categories are accessed:

- Master Control File
- Package Control File
- ACMQ root and XREF data sets
- Base/delta libraries **1**
- Source input libraries (for example, include libraries)
- Source output libraries (for example, load libraries) **1**
- All data sets contained in generate, move, or delete processors **1**
- CCID validation data set
- Package data set
- Package ship staging data sets
- Batch request data sets
- ISPF (foreground) data sets
- ACMQ data sets

**Notes:**

1. Access level of CONTROL is required for the Master Control File, package data set, and Endeavor LIB VSAM data sets. Access level of UPDATE is required for all others.
2. **1** — Alternate id support is not available if these are HFS Files.

**2.1.1.2 Data Sets Not Controlled by the Alternate User ID**

Several categories of data sets can only be accessed by the user's originating TSO user ID. These data sets include:

- Foreground ISPF Data Sets (such as *uid.C1TEMPRnMSGs*, *uidC1TEMPncntl*, *uid.C1#NTMPL.LIST*, etc.)
- ADD/UPDATE FROM: *data set*
- RETRIEVE TO: *data set*
- ARCHIVE/TRANSFER TO: *data set*
- COPY FROM: *data set*
- RESTORE FROM: *data set*
- LIST FROM: *data set*
- TRANSFER FROM: *data set*
- PRINT TO: *data set*
- Batch request data sets
- Endeavor\Lib *data sets* - when controlled by CA Common Services CA-L-Serv

**Note:** The alternate ID is **not** swapped on the creation or deletion of a data set or PDS in a processor. The user's TSO user ID must have authority to create/delete the data set or PDS.

## 2.2 Activating Data Set Protection

To activate data set protection, include the following list of parameters in the TYPE=MAIN section of the Endeavor Defaults Table (C1DEFLT5):

Parameter	Description
RACFGRP	The name of the RACF group with which the RACF user ID (RACFUID) is associated. This is an optional parameter, depending upon whether your site requires or elects this relationship.
RACFPWD	The password for the RACFUID defined. This is an optional parameter.
RACFUID	The RACF, eTrust CA-ACF2, or eTrust CA-Top Secret user ID to be used for data set authorization checking.

The following table shows the section of C1DEFLT5 containing the RACF parameters described above.

```

C          1          2          3          4          5          6          7 7
1          0          0          0          0          0          0          0 2

C1DEFLT5 TYPE=MAIN,
    ACCSTBL=NEWTNEQU,          ACCESS SECURITY TABLE          *
    APRVFLG=N,          APPROVAL PROCESSING (Y/N)          *
    ESSI=Y,          ESI ENABLED          *
    PKGCSEC=Y,          PACKAGE CAST SECURITY          *
    PKGSEC=ESI,          USE EXTERNAL SECURITY          *
    RACFGRP=,          E/OS390 ALT ID SUPPORT GROUP NAME          *
    RACFPWD=,          E/OS390 ALT ID SUPPORT PASSWORD          *
    RACFUID=,          E/OS390 ALT ID SUPPORT USERID          *

```

Once you have entered these parameters, assemble and link C1DEFLT5 into the Endeavor authorized load library and issue an LLA refresh for the library.

Because most passwords fall into an expiration life cycle, the RACFPWD must be monitored and maintained properly. Failure to do so may disable Endeavor if the password is changed or if it expires.

**Note:** In order to perform periodic maintenance against the various files under Endeavor control (for example, PDS compress, VSAM Repro), you must supply the RACFUID/RACFPWD or grant your Endeavor administrator the same permissions as the RACFUID. Also, RACFUID needs READ access to the CONLIB. Some types of file maintenance, such as a full file restore, may require a higher level of authority than that granted to the RACFUID. Be sure to allow someone the appropriate authority to address this contingency.

For information on configuring alternate ID support security options, consult the appropriate site security documentation.

## 2.3 Program Pathing Using eTrust CA-ACF2 and eTrust CA-Top Secret

Program pathing currently is available through both the *eTrust CA-ACF2* and *eTrust CA-Top Secret* security packages. Data set security is implemented through a special Endeavor alternate ID interface, using an assigned RACF user ID and optional password for Endeavor.

When using either *eTrust CA-ACF2* or *eTrust CA-Top Secret* with program pathing, you must define the top level programs that recognize Endeavor as the program in control for both foreground and batch processing. Resource rules **must** be written for the top-level programs. Depending on site requirements, resource rules may be written for other other Endeavor programs. A list of the Endeavor's top-level programs requiring resource rules follows:

NDVRC1	C1BM5000	BC1PNLST	ENBE1000
BC1PSRVL	C1BM6000	BC1PNCPY	ENBP1000
C1SM1000	C1BR1000	BC1POPEN	IEFIIC
C1BM3000	BC1PNLIB	IEBCOPY	ENDIE000

**Note:** This list can change. Your site security needs may require resource rules for other Endeavor programs, you can simply add these programs to the list above.

If you are a *eTrust CA-Top Secret* user, you can simplify the coding process by using the PRIVPGM option when writing a resource rule.

Data set security for *eTrust CA-ACF2* and *eTrust CA-Top Secret* requires optional APARS and USERMODS from the appropriate Computer Associates support organizations.

Endeavor runs as an ISPF dialog under program ISPTASK, as shown below. ISPTASK must be addressed from a global point across the entire TSO environment, prior to attempting program pathing for Endeavor or any other ISPF dialog package.



## **Chapter 3. Enabling Endeavor Native Security**

---

## 3.1 Native Security Tables

Endevor Native Security Tables allow you to protect the functional side of your system. Native security protects your environments, systems, and subsystems from unauthorized access. In addition, element names can be defined for use only within specific systems or subsystems.

You can use the Endevor External Security Interface (ESI) in lieu of native security if you want to integrate your existing security package with Endevor. Refer to Chapter 4, “Endevor External Security Interface (ESI)” for more information on Endevor ESI. For information on deciding whether Endevor ESI is an appropriate security option for your installation, refer to Chapter 1, “Security Overview.”

Endevor uses three tables to restrict access to functions and inventory levels:

- **Access Security Table** — Defines the environment(s) to which each user has access. There is one Access Security Table for each installed site. The Access Security Table must always be present in each site.
- **User Security Table** — Defines the systems/subsystems available to each user within a particular environment, and for each system/subsystem combination, the level of activity for which each user is permitted access (browse-only, delete, add, and so forth). There is one User Security Table for each environment.
- **Resource Security Table** — Defines any elements that are restricted to particular system(s)/subsystem(s), and for each restricted element, the type of action(s) for which the restriction applies. This table is defined separately for each environment. There is one Resource Security Table for each environment.

To prevent user access to specific systems/subsystems or environments, you must define the Access Security Table and the User Security Table.

To prevent unauthorized access to elements and actions, you must explicitly define the Resource Security Table.

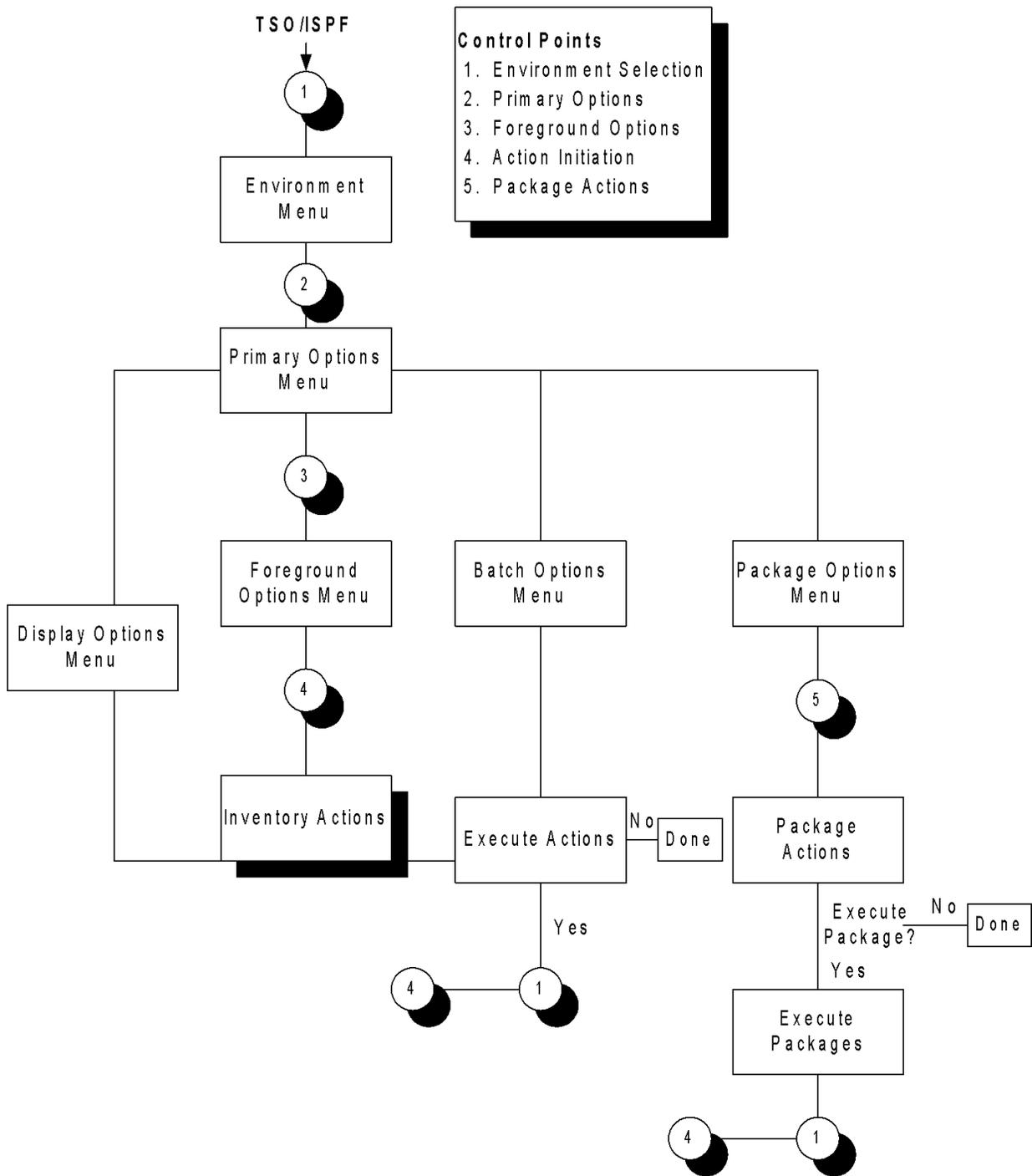
**Note:** Additional security checking is available through User Exit 1, described in the *Exits Guide*.

Native security tables are checked by the system at a series of security control points. The following sections describe the role of security control points in the Endevor processing flow.

## 3.2 Security Control Points

Security control within Endeavor is handled automatically through a series of security control points. Each control point invokes a routine that checks access and user privileges defined in the native security tables.

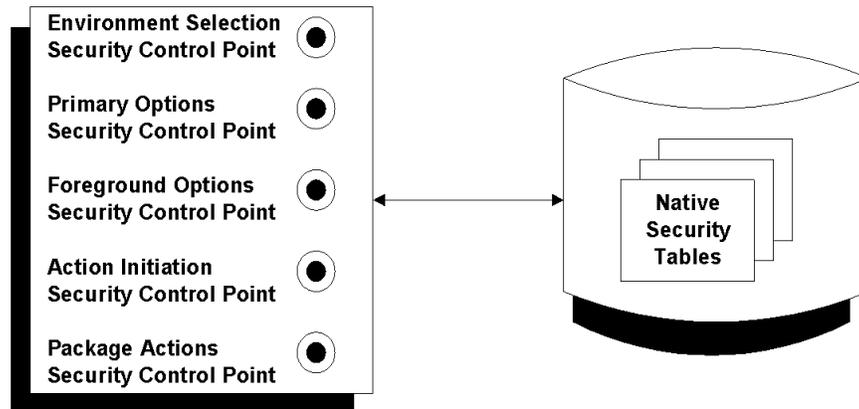
The following figure illustrates the strategic locations of security control points in the Endeavor processing flow. Each security control point governs access to functions and inventories occurring below it.



For example, the Environment Selection security control point controls access to the Environment menu; the Primary Options security control point controls access to the Primary Options menu.

Note the Environment Selection security control point appearing beneath the Batch Options menu. It regulates access to elements and functions available through the Batch Options and Package Options menus.

At each control point, Endeavor checks the native security tables to establish the access privileges defined for a particular user, environment, or system/subsystem.



### 3.2.1 Environment Selection Security Control Point

The Environment Selection security control point (also referred to as the Environment Access security control point) checks the Access Security Table to determine environment access privileges. This security check occurs prior to building the Environment Selection menu used to gain access to an environment.

### 3.2.2 Primary Options Security Control Point

The Primary Options security control point checks the User Security Table to determine a user's primary option privileges. These privileges include the Defaults, Display, Foreground, Batch, Environment, and Tutorial options. This security check occurs prior to building the Primary Options menu for an environment after access has been granted through the Environment Selection menu.

### 3.2.3 Foreground Options Security Control Point

The Foreground Options security control point checks the User Security Table to determine a user's foreground options. These privileges include Display, Add/Update, Retrieve, Generate, Move, Delete, Print, and Signin options. This security check occurs prior to building the Foreground Options menu for an environment.

### 3.2.4 Action Initiation Security Control Point

The Action Initiation security control point occurs:

- Prior to a cast operation during package processing, if the Defaults Table PKGCSEC flag is set to **Y**. Refer to the *Installation Guide* for more information on the PKGCSEC flag.
- During package verification processing.
- Prior to performing an Endeavor action.

The Action Initiation security control point checks the User Security Table to determine a user's action privileges as defined by the user's access level. These privileges include Endeavor actions, such as ADD, RETRIEVE, GENERATE, MOVE, DISPLAY, ARCHIVE, RESTORE, COPY, and LIST, as well as the SIGNOUT OVERRIDE, and ALL options.

### 3.2.5 Package Actions Security Control Point

The Package Actions security control point occurs prior to performing the requested package action.

**Note:** The actions used to create and use packages are not controlled by native security, but can be controlled by ESI.

## 3.3 User Exit Modules

You can define a user exit module at exit point 1 to do the following:

- Supplement the menu-building checks the system makes at each security control point.
- Supplement the action-request authorization that occurs at each security control point.

Exit 1 can only further restrict security, it cannot override restrictions imposed by your site security package. Refer to the *Exits Guide* for more information information about exits.

## 3.4 How Endeavor Reads the Security Tables

When verifying access to site components, Endeavor uses the first applicable entry it encounters in each security table.

### 3.4.1 User Access

To verify a user's access to an environment, Endeavor checks the Access Security Table for a userid entry that applies for that user. A match to the requesting user ID (user ID 7, for example) could take the form of a full mask (\$\$\$\$\$\$), a partial name (USER\$), or an exact match on the name (user ID 7) for the User Security and Resource Security tables. For the Access Security Table, you can use a single mask character (\$) to specify a full mask for system and subsystem security. Endeavor uses the most restrictive specification that matches the user ID to validate the access.

### 3.4.2 System or Subsystem Access

To verify a user's access to a particular system and subsystem, Endeavor checks the User Security Table for the first entry that applies for that user, again matching to a full mask (\$), a partial name (USER\$), or an exact match on the name (user ID 7). It uses the specifications defined by this entry to validate the access.

### 3.4.3 Resource Restrictions

To check for restrictions that apply when performing actions, Endeavor looks in the Resource Security Table for an entry that applies to the element specified for the current action (CAELE1, for example). The match by element (resource) name is based on a full mask (\$), a partial name (C\$), or an exact match to the name (CAELE1).

If Endeavor does not find a table entry for the element, processing continues. If it does find a table entry for the element, however, Endeavor checks to see if that entry includes the system and subsystem specified in the action, matching the table entry's system/subsystem name based on a full mask, a partial name, or an exact match. If the entry includes the system/subsystem for the action request, Endeavor allows the action if the access level required for the action is specified in the table entry.

If the appropriate access level is not specified **explicitly** in the table entry, Endeavor does not allow the action.

## 3.5 Implementing Native Security: A Quick Reference

To implement native security, you need to perform four basic steps:

1. Define the Access, User, and Resource Security Tables to control access to the environments, systems/subsystems, and elements you want to protect from unauthorized access.
2. Assemble and link-edit each security table you have defined.
3. Update the Defaults Table to reference the security tables you have defined.
4. Once you have sufficiently tested your functional security, copy the security table you defined as well as the new Defaults table, into an authorized library and perform an IPL or LLA refresh.

**Note:** If you are testing your security system, you do not need to issue an LLA refresh each time you edit a native security table if the tables are not in an authorized linklist library. Once you are ready to go into production mode, you need to copy your security tables into an authorized linklist library for safekeeping, and issue an LLA refresh.

Sections that follow explain each of these steps in greater detail.

### 3.5.1 Defining Endeavor Native Security Tables

To enable Endeavor native security, you must define the native security tables to control access to the inventories and functions you want to protect. Unless you do so, these functions are not protected.

To define a native security table, follow these steps:

1. Access the native security table you want to define.
2. Enter the appropriate CONSDEF macros.

After you have defined a security table, you need to:

1. Assemble and link-edit the table.
2. Update the Defaults Table.
3. Perform an LLA refresh if the table is in an authorized linklist library.

### 3.5.2 Entering CONSDEF Macros

You use a series of CONSDEF macros to define each security table. Once the CONSDEF macros have been assembled and link-edited successfully for each table, the table definition is complete. The table goes into effect once the Defaults Table is updated. Brief instructions for updating the Defaults Table appear in 3.9, “Modifying the Defaults Table” on page 3-27. For a complete description of this procedure, refer to the *Installation Guide*.

Before coding the CONSDEF macros, you should understand:

- How to use coding conventions.
- How to specify a mask.
- How Endeavor reads security definitions.

Refer to sections on defining each security table for more information on CONSDEF macro parameters.

**Note:** If a user is in Endeavor while a security table is being updated or edited, the administrator must exit and reenter Endeavor in order for the new table to take effect.

### 3.5.2.1 Coding Conventions

When coding CONSDEF macros, follow standard IBM rules:

- Place all macro specifications between columns 2 and 71 of the definition deck. To continue across cards, place an **X** in column 72 and start the continuation card in column 16.
- Include at least one space between the macro name, CONSDEF, and the first keyword parameter, TYPE. Do not include any more spaces. An exception applies for literal operands, where spaces can be included between the surrounding (single) quotes.
- Specify each macro keyword fully. You cannot use a shortened version of any keyword.

### 3.5.2.2 Order of Security Definitions

Endeavor uses the most restrictive parameters to determine security checks for each user (Access Security Table and User Security Table) and each resource (Resource Security Table).

In the following example, users having IDs that start with the letters CADEV, except for the two exceptions, have access to all systems and subsystems, for all types of processing.

```
C          1      1                      7 7
1          0      6                      0 2
```

```
CONSDEF TYPE=USER,USERID=CADEV1,SYSDEF=((FINANCE,$,N))
CONSDEF TYPE=USER,USERID=CADEV8,SYSDEF=((,$,$,N))
CONSDEF TYPE=USER,USERID=CADEV$,SYSDEF=((,$,$,ADMPRSUZ,BV))
```

The two exceptions, CADEV1 and CADEV8, are restricted in the following ways:

- User CADEV1 can access all systems except FINANCE.
- User CADEV8 cannot access any system.

## 3.6 Defining the Access Security Table

### 3.6.1 Overview

You use the Access Security Table to specify the environment(s) to which each user has access. You do this by specifying the unique Stage 1 name for the environment you want to access. There is one Access Security table for each installed site.

You define access to environment(s) either directly for each user ID, or for groups of users. Where you specify access for a group, each group must be associated with the specific user IDs included in the group (either before or after the group-level definition).

You might use the specification below, for example, to allow user DMS access to the stage names you specify; this allows user DMS access to the Stage 1 name in the environment in which it is located:

```
TYPE=USER,USERID=DMS,SYSDEF=((stage1-name,$,R),(stage1-name,$,R))
```

Or, you might permit access to the stage names you specify through group DVLP, then associate user DMS with that group:

```
TYPE=USER,GROUP=DVLP,SYSDEF=((stage1-name,$,R),(stage1-name,$,R))
TYPE=USER,USERID=DMS,GROUP=DVLP
```

To define the Access Security Table, enter CONSDEF macros in the following format:

```
C      1      1      7 7
1      0      6      0 2
```

```
CONSDEF TYPE=START, TABLE=USER
```

```
Form 1:
CONSDEF TYPE=USER,USERID=user-id,SYSDEF=((stage1-name,$,R)...)
:
```

```
Form 2:
CONSDEF TYPE=USER,GROUP=group-name,SYSDEF=((stage1-name,$,R)...)
:
```

```
Form 3:
CONSDEF TYPE=USER,USERID=user-id,GROUP=group-name
:
```

```
CONSDEF TYPE=END, TABLE=USER
```

You can use any combination of these statements, but must start with the TYPE=START macro and end with the TYPE=END macro.

**Note:** It is essential that you give each stage in each environment a unique name. Since you are specifying access to environments by way of stage name, you must be sure that each stage name is unique, across environments.

### 3.6.1.1 type=start,table=user

Defines the beginning of the Access Security Table definition.

### 3.6.1.2 type=end,table=user

Terminates the table definition.

### 3.6.1.3 type=user

Either defines the environment(s) accessible to a particular user or group (forms 1 and 2) or associates a user with a group-level access definition (form 3). The TYPE=USER macros can be coded in any order. Be sure, however, that each group associated with a user (form 3) has a corresponding group access definition (form 2).

A user must be given explicit permission through the TYPE=USER statements to process in an Endeavor environment.

## 3.6.2 type=user Parameters

Each TYPE=USER parameter is described below:

### 3.6.2.1 userid=userid

Defines the user(s) for which access information is being specified (form 1) or, if used with a GROUP specification (form 3), the ID of the user(s) associated with the *group-name*. The userid can end with a mask character to include all users having IDs that start with the characters specified or it can be specified as eight mask characters, \$\$\$\$\$\$\$ to include all users. The following parameter specifies all users having IDs that start with AN:

```
USERID=AN$
```

### 3.6.2.2 group=group-name

Defines the group (1-8 characters) for which access information is being specified (form 2) or, if used with a user ID specification (form 3), the group associated with the userid specified. Group names are specific to this macro; they are not referenced elsewhere within Endeavor. A group name might be PAYROLL, QA, DBA, SYSTEMS, and so forth. Once a group is defined, all users associated with the group acquire access to the environment(s) defined for the group.

### 3.6.2.3 sysdef=((stage1-name,\$,r))

Specifies an environment to which the user (form 1) or group (form 2) has access. The environment is defined in terms of the *stage1-name* defined for the environment. Computer Associates recommends using different stage names for each environment. The second and third positional parameters within the SYSDEF specification must be the characters \$ and R, respectively.

The mask character is supported for *stage1-name*. Generally, however, it is easier to repeat the operands within the parentheses when defining access to multiple environments, as shown below:

```
SYSDEF=((stage-1-name,$,R),(stage-1-name,$,R))
```

### 3.6.3 Assembling and Link-Editing the Access Security Table

Endevor supplies the JCL necessary to assemble and link-edit the Access Security Table during installation. It is placed in the `iprfx.igual.JCL` library as member `BC1JACCT`.

Once you have defined the Access Security Table, verify that the JCL is correct, then run the job to assemble and link-edit the new table. The name of the table is specified as the `SYSLMOD` member name in the JCL.

The output load module for the table is placed in the `LOADLIB` established during installation.

**Note:** For testing purposes, you can copy the Access Security Table to a standard load library. Once you are ready to go into production mode, be sure to copy the load module to an authorized `LINKLIST` library. Refer to the *Installation Guide* for a complete description of this procedure.

### 3.6.4 Update the Defaults Table

Following a successful link-edit, update the Defaults Table to point to the new Access Security Table. Set the `ACCSTBL` Defaults Table parameter in the `TYPE=MAIN` macro to identify the `SYSLMOD` DD member name assigned in the JCL.

See 3.9, “Modifying the Defaults Table” on page 3-27 for a brief description of this procedure. For a more complete treatment of this procedure, refer to the *Installation Guide*.

The sample JCL and Access Table Source in member `BC1JACCT` in the `iprfx.igual.JCL` library is shown below.

### 3.6 Defining the Access Security Table

---

```
//* ( COPY JOBCARD )
/*****
//*
//* BC1JACCT - BUILD THE ACCESS SECURITY TABLE FOR THE      *
//*           ENDEVOR ENVIRONMENT. THIS STEP IS USED      *
//*           TO DEFINE THE ACCESS TABLE FOR USE IN      *
//*           SETTING UP ENVIRONMENT SECURITY.             *
//*
//* STEP1 WILL ASSEMBLE THE MEMBER SPECIFIED.             *
//*
//* STEP2 WILL LINKEDIT THE MEMBER AND STORE IN USER LOADLIB *
//*   *** THE MODULE NAME MUST BE PUT IN THE DEFAULTS TABLE *** *
//*   *** THE MODULE NAME ACCSTABL CAN BE CHANGED ***      *
//*
/*****
//STEP1   EXEC PGM=ASMA90,PARM='NODECK,OBJECT,NOTERM'
//SYSLIB  DD  DISP=SHR,DSN=iprfx.iqua1.SOURCE
//        DD  DISP=SHR,DSN=SYS1.MACLIB
//SYSLIN  DD  DISP=(,PASS,DELETE),DSN=&&SYSLIN,
//           UNIT=tdisk,SPACE=(TRK,(3,5),RLSE),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSPUNCH DD DUMMY
//SYSUT1  DD  UNIT=tdisk,SPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*
//SYSIN   DD  *
           CONSDEF TYPE=START,TABLE=USER
           CONSDEF TYPE=USER,GROUP=ADMIN,                X
                 SYSDEF=((STG1,$,R),(FIN1,$,R),(TESTSTG1,$,R))
           CONSDEF TYPE=USER,GROUP=FINANCE,              X
                 SYSDEF=((STG1,$,R),
                       (FIN1,$,R))
           CONSDEF TYPE=USER,GROUP=APPL,                 X
                 SYSDEF=((TESTSTG1,$,R))
           CONSDEF TYPE=USER,USERID=CAUID1$,GROUP=ADMIN
           CONSDEF TYPE=USER,USERID=CAUID2$,GROUP=FINANCE
           CONSDEF TYPE=USER,USERID=CAUID$,GROUP=APPL
           CONSDEF TYPE=END,TABLE=USER

/*
/*
//STEP2   EXEC PGM=IEWL,PARM='LIST,NCAL,RENT,XREF,SIZE=(256K,64K)',
//           COND=(0,NE)
//SYSLIN  DD  DISP=(OLD,DELETE),DSN=&&SYSLIN
/* MEMBER NAME CHANGE MUST ACCOMPANY CHANGE TO C1DEFLT5 FILE
//SYSLMOD DD  DISP=SHR,DSN=uprfx.uqua1.LOADLIB(ACCSTABL)
//SYSUT1  DD  UNIT=tdisk,SPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*
```

## 3.7 Defining the User Security Table

The User Security Table specifies the system(s) and subsystem(s) to which each user has access, within a particular environment. For each system/subsystem to which a user has access, the table also specifies the type(s) of processing allowed (retrieve-only, add, update, and so forth). There is (at most) one User Security Table for each environment.

You can define access to the system(s)/subsystem(s) either directly for each userid or for groups of users. Where you specify access at the group level, each group must be associated with the specific user IDs included in the group.

You might use the specification below, for example, to allow user DMS access to the Finance and GL systems (all subsystems). The user can access the Finance system to move (**M**), generate (**P**), or display (**B**) elements, and the GL system to display only.

```
TYPE=USER,USERID=DMS,SYSDEF=((FINANCE,$,MP,B),(GL,$,,B))
```

Alternatively, you might permit this access through group ACCT, then associate user DMS with that group:

```
TYPE=USER,GROUP=ACCT,SYSDEF=((FINANCE,$,MP,B),(GL,$,,B))
TYPE=USER,USERID=DMS,GROUP=ACCT
```

To define the User Security Table, enter CONSDEF macros using the format below:

```
C      1      1      7 7
1      0      6      0 2
```

```
CONSDEF TYPE=START, TABLE=USER
```

Form 1:

```
CONSDEF TYPE=USER,USERID=user-id,[UNTIL=yyddd,] X
        SYSDEF=((sys-name,subsys-name,access1,access2)...
```

⋮

Form 2:

```
CONSDEF TYPE=USER,GROUP=group-name,[UNTIL=yyddd,] X
        SYSDEF=((sys-name,subsys-name,access1,access2)...
```

⋮

Form 3:

```
CONSDEF TYPE=USER,USERID=user-id,GROUP=group-name
```

⋮

```
CONSDEF TYPE=END, TABLE=USER
```

You can use any combination of these statements, but must start with the TYPE=START macro and end with the TYPE=END macro.

### 3.7.1.1 type=start,table=user

Defines the beginning of the User Security Table definition.

### 3.7.1.2 type=end,table=user

Terminates the table definition.

### 3.7.1.3 type=user

Either defines the system(s)/subsystem(s) accessible to a particular user or group (forms 1 and 2) or associates a user with a group-level access definition (form 3). The type=user macros can be coded in any order. Make sure, however, that each group associated with a user (form 3) has a corresponding group access definition (form 2).

In order for a user to have access to a particular system/subsystem configuration, that user must be given explicit permission to process within that configuration, for the type of access desired.

## 3.7.2 type=user Parameters

Each TYPE=USER parameter is described below:

### 3.7.2.1 userid=user-id

Defines the user(s) for which access information is being specified (form 1), or, if used with a group specification (form 3), the ID of the user(s) associated with the *group-name*. The *user-id* can end with a mask character to include all users having IDs that start with the characters specified, or it can be specified as eight mask characters, \$\$\$\$\$\$\$\$, to include all users. The following parameter specifies all users with IDs that start with AN:

```
:USERID=AN$
```

### 3.7.2.2 group=group-name

Defines the group (1-8 characters) for which access information is being specified (form 2) or, if used with a user ID specification (form 3), the group associated with the *user-id* specified. Group names are specific to this macro; they are not referenced elsewhere within Endeavor. A group name might be Payroll, QA, DBA, Systems, and so forth. Once a group is defined, all users associated with the group acquire access to the configuration(s) defined for the group.

### 3.7.2.3 [until=yyddd]

Defines the (Julian) date through which the defined access is valid. This parameter is optional. If omitted, there is no expiration on the definition.

### 3.7.2.4 sysdef=( (sys-name,subsys-name,access1,access2)...)

Specifies a system/subsystem/access-level configuration to which the user (form 1) or group (form 2) has access. The configuration is defined using four positional parameters:

#### *sys-name*

The Endeavor system to which the user has access (restricted according to the *subsys-name* and access codes specified).

#### *subsys-name*

The name of the subsystem to which the user has access within the system named as option 1 (and restricted according to the access codes).

#### *access1*

A list of single-character codes that identify the types of access for which the user is authorized within the system and subsystem named. Do not include a separator character between multiple codes within each access specification. The *access1* parameter specifies all types of access except Display and Archive, which must be specified using *access2*. If you want to specify Display or Archive access only, include a positional comma for *access1*. If you want neither Display nor Archive access, you need not include a final positional comma.

#### *access2*

A list of single-character codes that identify the types of access for which the user is authorized within the system and subsystem named. Specify Display and Archive using *access2*.

Access codes for the *access1* and *access2* variables are listed below.

<b>Access Code</b>	<i>access1</i>	<i>access2</i>	<b>Access Level</b>
A	x		Add
B		x	Display
D	x		Delete
M	x		Move
P	x		Generate
R	x		Retrieve
S	x		Signout override
U	x		Update
V		x	Archive

<b>Access Code</b>	<i>access1</i>	<i>access2</i>	<b>Access Level</b>
Z	x		Environment administration
N	x		No access

The *sys-name* and/or *subsys-name* variables can end with a mask character (for example, DEV\$), to include all systems/subsystems starting with the same characters; or you can use a single mask character to include all systems/subsystems. For example, the following statement specifies add-access for all subsystems within systems beginning with the characters GL:

```
SYSDEF=((GL$, $, A))
```

To allow access to multiple configurations, repeat the operands within the parentheses:

```
SYSDEF=((sys,subsys,access,access), (sys,subsys,access,access) . . .)
```

The following table describes the access levels necessary for various types of Endeavor processing. Note that access levels for moves and transfers appear in a separate table that follows this table.

<b>Endeavor Action</b>	<b>Access Level</b>	<b>Access Code</b>
Add	Add	A
Update	Update	U
Retrieve	Retrieve	R
Generate (Stage 1)	Generate	P
Generate (Stage 2)	Move	M
Display	Retrieve or Display	R or B
Delete (Stage 1)	Delete	D
Delete (Stage 2)	Move	M
Signin	Retrieve	R
Print	Retrieve or Display	R or B
Archive	Archive	V
Restore to Stage 1	Add	A
Restore to Stage 2	Move	M
Copy	none	
List	Display	B
Signout override	Signout Override	S
All (element type Process)	Environment management	Z

<b>Endevor Action</b>	<b>Access Level</b>	<b>Access Code</b>
Checks while building the Primary Options menu:		
Include Defaults option	None	
Include Display option	Retrieve or Display	R or B
Include Foreground option	Retrieve	R
Include Batch option	Retrieve	R
Include Environment option	Environment management	Z
Include Tutorial option	None	
Checks while building the Foreground Options menu:		
Include Option	none	—
Include Add/Update option	Add or Update	A or U
Include Retrieve option	Retrieve	R
Include Generate option	Generate	P
Include Delete option	Delete	D
Include Print option	Retrieve or Display	R or B
Include Signin option	Retrieve	R

Moves and transfers often involve combined actions and source and target considerations that require special access codes. The following table provides the access codes required for performing moves and transfers according to stage, source, and target parameters.

<b>Endevor Action</b>	<b>Level of Access</b>	<b>Access Code</b>
Moves From:		
Stage 1	Move	M
Stage 2	Move	M
Moves To:		
Stage 1	Add	A
Stage 2	Move	M
Transfers From:		
Stage 1, no Delete	Retrieve	R
Stage 1 with Delete	Delete	D

Endevor Action	Level of Access	Access Code
Stage 2, no Delete	Retrieve	R
Stage 2 with Delete	Move	M
Transfers To: <b>1</b>		
Stage 1	Add	A
Stage 2	Move	M

**Note:**

**1** : Transfers to a target destination do not permit a delete.

### 3.7.3 Assembling and Link-Editing the User Security Table

Endevor supplies the JCL necessary to assemble and link-edit the User Security Table. It is placed in the iprfx.igual.JCL library as member BC1JUSRT.

Once you have defined the User Security Table, verify that the JCL is correct, then run the job to assemble and link-edit the new table. The name of the table is specified as the SYSLMOD member name in the JCL.

The output load module for the table is placed in the LOADLIB established during installation.

**Note:** For testing purposes, you can copy the User Security Table to a standard load library. Once you are ready to go into production mode, be sure to copy the load module to an authorized LINKLIST library. Refer to the *Installation Guide* for a complete description of this procedure.

Following a successful link-edit, update the Defaults Table (C1DEFULTS) to point to the new User Security Table. Set the USERTBL parameter in the TYPE=ENVRNMNT macro to identify the SYSLMOD DD member name assigned in the JCL.

See 3.9, “Modifying the Defaults Table” on page 3-27 for a brief description of this procedure. For a more complete treatment of this procedure, refer to the *Installation Guide*.

The sample JCL and User Table Source in member BC1JUSRT in the iprfx.igual.JCL library is shown below.

```

/** ( COPY JOBCARD )
/*******
/** BC1JUSRT - BUILD THE USER SECURITY TABLE FOR ENDEVOR *
/** ENVIRONMENT. THIS STEP IS USED TO DEFINE THE USER *
/** TABLE FOR USE IN SETTING UP SECURITY FOR THE *
/** ENVIRONMENT *
/** STEP1 WILL ASSEMBLE THE MEMBER SPECIFIED *
/** STEP2 WILL LINKEDIT THE MEMBER AND STORE IN USER LOADLIB *
/** *** THE MODULE NAME MUST BE PUT IN THE DEFAULTS TABLE *** *
/** *** THE MODULE NAME USERTABL CAN BE CHANGED *** *
/*******
//STEP1 EXEC PGM=ASMA90,PARM='NODECK,OBJECT,NOTERM'
//SYSLIB DD DISP=SHR,DSN=iprfx.iqual.SOURCE
// DD DISP=SHR,DSN=SYS1.MACLIB
//SYSLIN DD DISP=(,PASS,DELETE),DSN=&&SYSLIN,
// UNIT=tdisk,SPACE=(TRK,(3,5),RLSE),
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSPUNCH DD DUMMY
//SYSUT1 DD UNIT=tdisk,SPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
        CONSDEF TYPE=START,TABLE=USER
        CONSDEF TYPE=USER,GROUP=ADMIN, X
            SYSDEF=(($,$,AURMDPSZ,BV))
        CONSDEF TYPE=USER,GROUP=FINANCE, X
            SYSDEF=(($,$,AURMDPS,B))
        CONSDEF TYPE=USER,GROUP=APPL, X
            SYSDEF=((PERSONL,$,AURDPS,B),(RETIRE,$,AURDPS,B),
            (PAYROLL,$,AURDPS,B))
        CONSDEF TYPE=USER,USERID=CAUID1$,GROUP=ADMIN
        CONSDEF TYPE=USER,USERID=CAUID2$,GROUP=FINANCE
        CONSDEF TYPE=USER,USERID=CAUID3$,GROUP=APPL
        CONSDEF TYPE=USER,USERID=CAUID4$,GROUP=APPL
        CONSDEF TYPE=END,TABLE=USER

/*
//STEP2 EXEC PGM=IEWL,PARM='LIST,NCAL,RENT,XREF,SIZE=(256K,64K)',
// COND=(0,NE)
//SYSLIN DD DISP=(OLD,DELETE),DSN=&&SYSLIN
/** MEMBER NAME CHANGE MUST ACCOMPANY CHANGE TO C1DEFLT5 FILE
//SYSLMOD DD DISP=SHR,DSN=uprfx.uqual.LOADLIB(USERTABL)
//SYSUT1 DD UNIT=tdisk,SPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*

```

## 3.8 Defining the Resource Security Table

The Resource Security Table defines element names that are restricted to a particular system(s)/subsystem(s), within a particular environment. For each restricted element name, the table also specifies the type(s) of processing for which the restriction applies (retrieve-only, add, update, and so forth). There is one Resource Security Table for each environment.

You can define resource security directly for each resource (element name), or you can define it for groups of resources. Where you specify security at the group level, each group must be associated with the specific element names included in the group.

For example, you would use the specification below to indicate that any element names starting with the characters ACCT can only be added to the ACCOUNTS subsystem of the FINANCE system:

```
TYPE=RESOURCE,RNAME=ACCT$,SYSDEF=((FINANCE,ACCOUNTS,A))
```

This does not limit the ACCOUNTS subsystem (FINANCE system) to element names starting with the characters ACCT, but simply prevents these element names from being added to other systems/subsystems.

To illustrate a group-level specification, you would use the definition below to indicate that element names beginning with the characters AP or AR can only be added to the FINANCE or GL systems, any subsystem:

```
TYPE=RESOURCE,GROUP=XYZ,SYSDEF=((FINANCE,$,A),(GL,$,A))
TYPE=RESOURCE,RNAME=AP$,GROUP=XYZ
TYPE=RESOURCE,RNAME=AR$,GROUP=XYZ
```

To define the Resource Security Table, enter CONSDEF macros in the following format:

```
C          1      1                      7 7
1          0      6                      0 2
```

```
CONSDEF TYPE=START, TABLE=RESOURCE
```

Form 1:

```
CONSDEF TYPE=RESOURCE, RNAME=element-name, [UNTIL=yyddd,]      X
        SYSDEF=((sys-name,subsys-name,access1,access2)...) 
```

⋮

Form 2:

```
CONSDEF TYPE=RESOURCE, GROUP=group-name, [UNTIL=yyddd,]      X
        SYSDEF=((sys-name,subsys-name,access1,access2)...) 
```

Form 3:

```
CONSDEF TYPE=RESOURCE, RNAME=element-name, GROUP=group-name
```

⋮

```
CONSDEF TYPE=END, TABLE=RESOURCE
```

You can use any combination of these statements, but must start with the TYPE=START macro and end with the TYPE=END macro.

### 3.8.1.1 `type=start,table=resource`

Defines this as the beginning of the Resource Security Table definition.

### 3.8.1.2 `type=end,table=resource`

Terminates the table definition.

### 3.8.1.3 `type=resource`

Defines an element name(s) that is restricted to a particular system and subsystem (form 1) or a group of element names that are restricted to a particular system and subsystem (form 2) or associates an element name with a group-level definition (form 3). The TYPE=RESOURCE macros can be coded in any order. Make sure, however, that each group associated with an element name (form 3) has a corresponding group definition (form 2).

In order for a restriction to apply for an element name, that name must be associated with the type(s) of processing (access levels) for which the restriction applies (retrieve-only, add, update, and so forth).

## 3.8.2 `type=user` Parameters

Each TYPE=RESOURCE parameter is described below:

### 3.8.2.1 `rname=element-name`

Defines the element name(s) for which a restriction is being specified (form 1) or, if used with a group specification (form 3), the element name(s) associated with the *group-name*. The *element-name* can end with a single mask character to include all element names that start with the characters specified, or it can be specified with eight mask characters, (\$\$\$\$\$\$\$), to include all element names. The following parameter specifies all element names that start with the characters FIN:

```
RNAME=FIN$
```

### 3.8.2.2 `group=group-name`

Defines the group (1-8 characters) for which a restriction is being defined (form 2), or, if used with an RNAME specification (form 3), the group associated with the *element-name* specified. Group names are specific to this macro; they are not referenced elsewhere within Endeavor. A group name might be Payroll, QA, DBA, Systems, and so forth. Once a group is defined, all element names associated with the group acquire the restrictions defined for the group.

### 3.8.2.3 [until=yyddd]

Defines the (Julian) date through which the defined restriction is valid. This parameter is optional. If omitted, there is no expiration on the definition.

### 3.8.2.4 sysdef=( (sys-name,subsys-name,access1,access2)...) )

Specifies the system/subsystem configuration to which the element name (form 1) or group (form 2) is restricted. The configuration is defined using four positional parameters:

*sys-name*

The name of the Endeavor system to which the element name is restricted. You can use a single mask character (\$) to specify all system names.

*subsys-name*

The name of the subsystem to which the element name is restricted, within the system named. You can use a single mask character (\$) to specify all subsystem names.

*access1*

A list of single-character codes that identify the types of access under which the element name is restricted, within the system and subsystem named. Within each access specification, do not include a separator character between multiple codes. The parameter *access1*< specifies all types of access except Display and Archive, which must be specified using *access2*.

*access2*

A list of single-character codes that identify the types of access under which the element name is restricted, within the system and subsystem named.

If you want to specify Display or Archive access only, include a positional comma for *access1*. If you want neither Display nor Archive access, you need not include a final positional comma.

Access codes are listed in the tables in Defining the User Security Table, earlier in this chapter.

The *sys-name* and/or *subsys-name* variables can end with a mask character (for example DEV\$) to include all systems/subsystems that start with the characters specified, or they can be specified as a single mask character to include all systems/subsystems.

To restrict element names to any of several configurations, repeat the operands within the parentheses:

```
SYSDEF=((sys,subsys,access,access),(sys,subsys,access,access)...) )
```

### 3.8.3 Assembling and Link-Editing the Resource Security Table

Endevor supplies the JCL necessary to assemble and link-edit the Resource Security Table. It is placed in the JCL library as member BC1JRSCT during installation.

Once you have defined a Resource Security Table, verify that the JCL is correct, then run the job to assemble and link-edit the new table. The name of the table is specified as the SYSLMOD member name in the JCL.

The output load module for the table is placed in the LOADLIB established during installation.

**Note:** For testing purposes, you can copy the Resource Security Table to a standard load library. Once you are ready to go into production mode, be sure to copy the load module to an authorized LINKLIST library. Refer to the *Installation Guide* for a complete description of this procedure.

Following a successful link-edit, update the Defaults Table to point to the new Resource Security Table. Set the RSCETBL parameter in the TYPE=ENVNMNT macro to identify the SYSLMOD DD member name assigned in the JCL.

Refer to 3.9, “Modifying the Defaults Table” on page 3-27 for a brief description of this procedure. For a more complete treatment of this procedure, refer to the *Installation Guide*.

The following figure shows the sample JCL and Resource Table Source in member BC1JRSCT in the iprfx.igual.JCL library.

### 3.8 Defining the Resource Security Table

---

```
//* ( COPY JOBCARD )
/*****
//*
//* BC1JRSCT - BUILD THE RESOURCE TABLES FOR ENDEVOR          *
//*              ENVIRONMENT. THIS STEP IS USED TO DEFINE THE  *
//*              RESOURCE TABLE FOR USE IN SETTING UP SECURITY *
//*              FOR THE ENVIRONMENT                            *
//*
//* STEP1 WILL ASSEMBLE THE MEMBER SPECIFIED                   *
//*
//* STEP2 WILL LINKEDIT THE MEMBER AND STORE IN USER LOADLIB  *
//*   *** THE MODULE NAME MUST BE PUT IN THE DEFAULTS TABLE *** *
//*   *** THE MODULE HERE RSCTTABL CAN BE CHANGED ***         *
//*
/*****
//STEP1   EXEC PGM=ASMA90,PARM='NODECK,OBJECT,NOTERM'
//SYSLIB  DD  DISP=SHR,DSN=iprfx.iqua1.SOURCE
//        DD  DISP=SHR,DSN=SYS1.MACLIB
//SYSLIN  DD  DISP=(,PASS,DELETE),DSN=&&SYSLIN,
//           UNIT=tdisk,SPACE=(TRK,(3,5),RLSE),
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSPUNCH DD DUMMY
//SYSUT1  DD  UNIT=tdisk,SPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*
//SYSIN   DD  *
           CONSDEF TYPE=START, TABLE=RESOURCE
           CONSDEF TYPE=RESOURCE, RNAME=BNVJ$,                X
                 SYSDEF=((NDVR8612,BETAJCL,AURMDPSZ))
           CONSDEF TYPE=RESOURCE, RNAME=PER$,                  X
                 SYSDEF=((PERSONL,$,AURMDPSZ))
           CONSDEF TYPE=RESOURCE, RNAME=$$$$$$,                X
                 SYSDEF=((,$,$,R))
           CONSDEF TYPE=END, TABLE=RESOURCE
/*
//STEP2   EXEC PGM=IEWL,PARM='LIST,NCAL,RENT,XREF,SIZE=(256K,64K)',
//        COND=(0,NE)
//SYSLIN  DD  DISP=(OLD,DELETE),DSN=&&SYSLIN
/* MEMBER NAME CHANGE MUST ACCOMPANY CHANGE TO C1DEFLT5 FILE
//SYSLMOD DD  DISP=SHR,DSN=uprfx.uqua1.LOADLIB(RSCTTABL)
//SYSUT1  DD  UNIT=tdisk,SPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*
```

## 3.9 Modifying the Defaults Table

To enable native security, you need to:

- Modify the Defaults Table to reference the security tables you have defined.
- Copy the Defaults Table into an authorized load library. This step should be undertaken only *after* you have completed testing your security definitions.
- Perform an LLA refresh.

### 3.9.1 Procedure

To modify the Defaults Table, perform the following steps:

1. Copy the C1DEFLT macro found in member DEMODEFT in the DEMOSRC library into the current Endeavor Defaults Table source.
2. Specify the following parameters in the Defaults Table.

**TYPE=MAIN Parameters**

**ACCSTBL**

A 1- to 8-character name of the Access Security Table in use at your site.

**TYPE=ENVRMNT: Parameters:**

**USERTBL**

A 1- to 8-character name of the User Security Table for this environment.

**RSCETBL**

A 1- to 8-character name of the Resource Security Table for this environment.

3. Run job BC1JDEFT again, to reassemble and relink-edit the C1DEFLT macros, placing the output Defaults Table in the Endeavor LOADLIB (member name C1DEFLT).

Refer to the *Installation Guide* for more information on updating and modifying the Defaults Table.

For more information about load libraries, refer to the *Administration Guide*.



## **Chapter 4. Endeavor External Security Interface (ESI)**

---

## 4.1 Endeavor ESI Overview

The Endeavor External Security Interface (ESI) is an Endeavor option that unifies security for Endeavor and your site security package (RACF, *eTrust CA-ACF2*, or *eTrust CA-Top Secret*). The ESI option enables you to:

- Extend your site security package to control and authorize access to components maintained by Endeavor.
- Secure user actions ranging from environment access to specific action checks.
- Customize ESI security through a table-driven architecture.

ESI converts combinations of Endeavor entities such as an environment name or an element name into pseudo data set names and then queries your site security package for a ruling whether or not the user is allowed to access the data set. Examples of Endeavor entities that you can map into your site security package rules are:

- Environments
- Elements
- Stages
- CCIDs
- Actions
- Systems/subsystems

**Note:** Each node in a data set name only allows a maximum of eight characters per value; therefore, values greater than eight characters are truncated to eight characters (e.g., EMERGENC).

### 4.1.1 Security Checkpoints

ESI uses Endeavor provided security checkpoints (Exit 01) to determine whether to allow or deny a user access to an inventory, environment or a specific function/action. ESI processing is invoked at all security control points. See Chapter 1, “Security Overview” for an introduction to the five Endeavor security control points.

### 4.1.2 How ESI Security Works

ESI uses a user compiled table called the Name Equates Table to specify rules that map Endeavor entity names to pseudo data set names when a security control point is encountered. ESI constructs the pseudo data set name and executes the operating system's RACROUTE macro. The RACROUTE macro provides an application with access to the System Authorization Facility (SAF) which in turn communicates with your site security package (RACF, *eTrust CA-ACF2*, or *eTrust CA-Top Secret*). SAF allows Endeavor to request authorization information for any site security package.

The Name Equates Table consists of a set of ESI Defaults entries (ESIDFLTS), Function Equates entries (FUNCEQU) and Name Equates entries (NAMEQU) described in the following sections.

#### 4.1.2.1 ESI Defaults Entries (ESIDFLTS)

The ESIDFLTS entry allows you to establish the default behavior for all security call formats and tracing. It also includes an option to improve performance. See the ESIDFLTS description in 4.2.1, “Defining ESI Diagnostics” on page 4-8.

#### 4.1.2.2 Function Equates Entries (FUNCEQU)

Each FUNCEQU entry allows you to map Endeavor actions to authorization values or keywords that are understood by your site security package. See 4.2.2.1, “Mapping Authorization Values to RACF, eTrust CA-ACF2, and eTrust CA-Top Secret” on page 4-11 for more information.

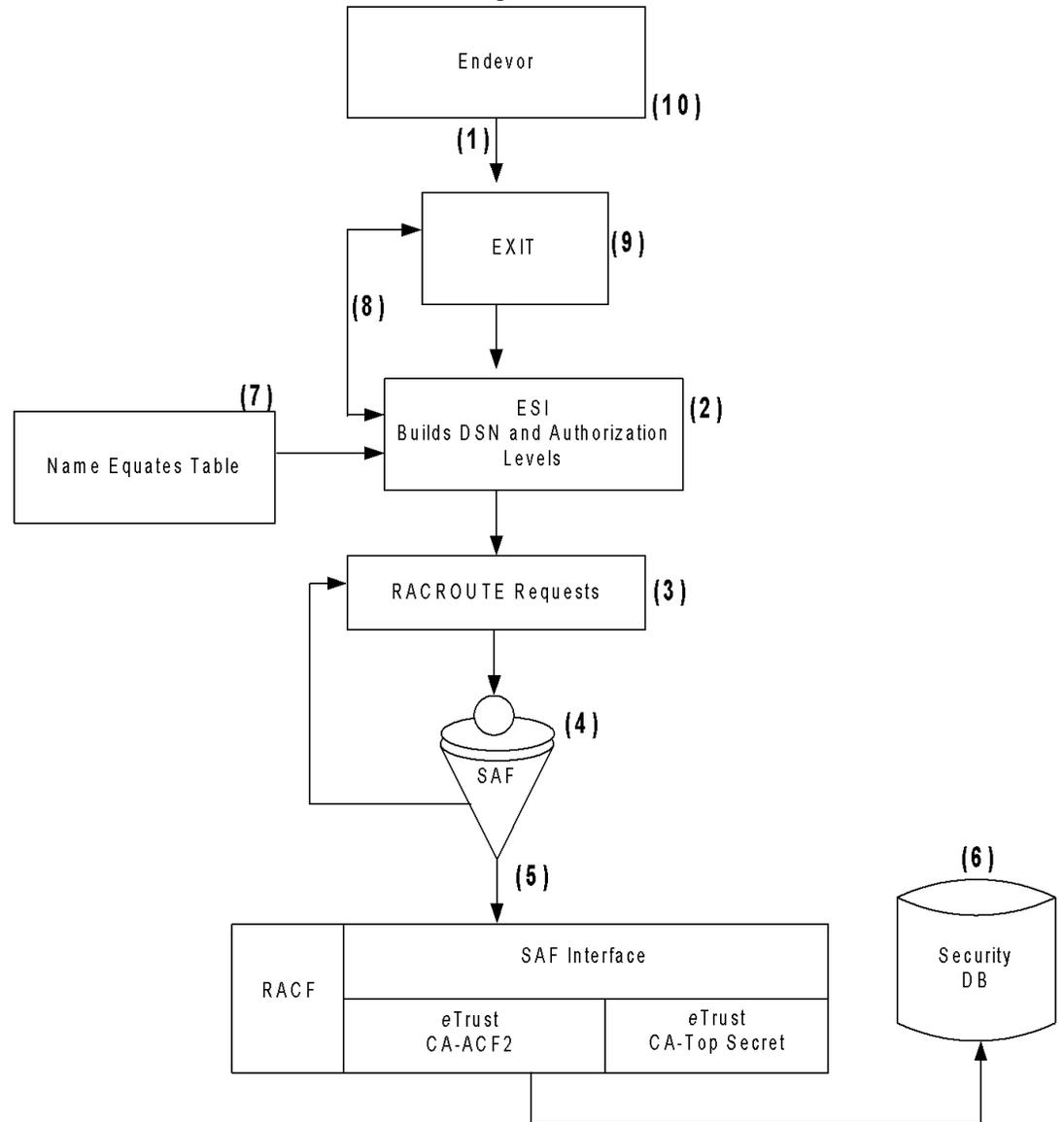
#### 4.1.2.3 Name Equates Entries (NAMEQU)

The NAMEQU entries become the pseudo data set names that are sent to the RACROUTE macro. Each entry corresponds to a specific security control point that you can secure. There are six different security call formats for pseudo data set. The correlation of the five pseudo data set formats to security control points are shown below.

Security Control Point	Format	Where the Security Check Occurs
Environment	ENVIRONMENT_ACCESS (formerly FORMAT1)	<ul style="list-style-type: none"> <li>■ Prior to building the Environment Selection menu.</li> <li>■ When you change the current environment through a panel.</li> <li>■ During batch processing to validate the user's access to the environment.</li> </ul>
Primary Options	PRIMARY_OPTIONS (formerly FORMAT2)	<ul style="list-style-type: none"> <li>■ Prior to building the Environments Primary Options menu.</li> <li>■ During UNLOAD/RELOAD processing.</li> <li>■ During batch package processing.</li> </ul>
Foreground Options	FOREGROUND_OPTIONS (formerly FORMAT3)	<ul style="list-style-type: none"> <li>■ Prior to building the Foreground Options menu for the environment.</li> </ul>
Action Initiation	ACTION_INITIATION (standard and extension) (formerly FORMAT4 and FORMAT5)	<ul style="list-style-type: none"> <li>■ Prior to performing the requested action.</li> <li>■ Prior to cast, during package processing if PKGCSEC=Y.</li> <li>■ During package verification processing.</li> </ul>
Package Actions	PACKAGE_ACTIONS	<ul style="list-style-type: none"> <li>■ Prior to performing the requested package action.</li> </ul>

#### 4.1.2.4 The Security Processing Model

The following figure shows how Endeavor ESI works with a site security package such as RACF, eTrust CA-ACF2, or eTrust CA-Top Secret.



The following list describes the security processing model illustrated in the previous figure.

1. Endeavor calls Exit 01 processing.
2. Exit 01 processing calls ESI.
3. ESI constructs the data set name (DSN) and authorization level from the Name Equates Table.

4. ESI issues a RACROUTE request with the DSN and authorization levels. RACROUTE requests are then routed to the System Authorization Facility (SAF).
5. SAF routes the RACROUTE request directly to the site security package.
6. The security package interprets the request by looking up the request in the security database.
7. SAF returns to ESI and the request either passes or fails.
8. ESI returns to Exit 01 processing.
9. Endeavor Exit 01 processing runs other user exits.
10. Exit 01 processing may return to Endeavor.

### 4.1.3 User Exit Modules

You can define a user exit module at exit point 1 to do the following:

- Supplement the menu-building checks the system makes at each security control point.
- Supplement the action-request authorization that occurs at each security control point.

**Note:** Exit 01 can only further restrict security, it cannot override restrictions imposed by your site security package (RACF, *eTrust CA-ACF2*, or *eTrust CA-Top Secret*). See the *Administration Guide* for more information about exits.

## 4.2 The Name Equates Table

The Name Equates Table allows you to specify security rules that map Endeavor entity names to pseudo data set names. A sample Name Equates Table is shown below:

```

C      1      2      3      4      5      6      7 7
1      0      0      0      0      0      0      0 2
      TITLE 'BC1TNEQU - EXTERNAL SECURITY INTERFACE TABLE.'
*****
*      DEFINE ESI DEFAULTS      *
*****
BC1TNEQU ESIDFLTS WARN=NO,      +
      TITLE='BC1TNEQU SECURITY INTERFACE TABLE',      +
      HEADER=YES,      +
      LATSIZ=2,      +
      DESC=6,      +
      ROUTCDE=11
*****
*      MAP E/OS390 AUTHORITIES TO SAF AUTHORITIES FOR      *
*      ACTION_INITIATION AND PACKAGE_ACTIONS FORMAT CALLS.      *
*      NOTE: ENVIRONMENT_ACCESS, PRIMARY_OPTIONS AND      *
*      FOREGROUND_OPTIONS FORMAT CALLS ALWAYS USE READ      *
*      AUTHORITY AND CANNOT BE MODIFIED.      *
*****
      FUNCEQU SAFAUTH=READ,      +
      C1ACTNS=(ADD,ARCHIVE,DELETE,      +
      DISPLAY,ENVRNMGR,GENERATE,MOVE,      +
      PBACKOUT,PCAST,PCCOMMIT,PCCREATE,PDISPLAY,      +
      PEXECUTE,PLIST,PMODIFY,PREVIEW,      +
      PUTILITY,RESTORE,RETRIEVE,      +
      SIGNIN,SIGNOVR,TRANSFER,UPDATE)
*****
*      SAMPLE SYNTAX OF OTHER SUPPORTED FUNCEQU AUTHORITIES LEVELS      *
*****
*      FUNCEQU SAFAUTH=NONE,      +
*      FUNCEQU SAFAUTH=UPDATE,      +
*      FUNCEQU SAFAUTH=CONTROL,      +
*      FUNCEQU SAFAUTH=ALTER,      +
*****
*      END OF FUNCEQU SECTION      *
*****
      FUNCEQU TYPE=END
      SPACE 2
*****
*      SPECIFY SAF DATASET NAME FORMATS      *
*****
      NAMEQU ENVIRONMENT_ACCESS,      +
      L1=('C1'),      +
      L2=('ENVIRON'),      +
      L3=(ENVIRONMENT)
      NAMEQU PRIMARY_OPTIONS,      +
      L1=('C1'),      +

```

```

                L2=(ENVIRONMENT),          +
                L3=('PMENU'),              +
                L4=(MENUITEM)
NAMEQU FOREGROUND_OPTIONS,                +
                L1=('C1'),                 +
                L2=(ENVIRONMENT),          +
                L3=('FORACTN'),           +
                L4=(MENUITEM)
NAMEQU ACTION_INITIATION,                 +
                L1=('C1'),                 +
                L2=(ENVIRONMENT),          +
                L3=(SYSTEM),               +
                L4=(SUBSYSTEM)
NAMEQU ACTION_INITIATION,                 +
                L1=('C1'),                 +
                L2=(MENUAUTH)
NAMEQU PACKAGE_ACTIONS,                  +
                L1=('C1'),                 +
                L2=('PACKAGE'),            +
                L3=(MENUITEM),            +
                L4=(PKGSUBFC),            +
                L5=(PKGID)
*****
*      SAMPLE SYNTAX OF OTHER SUPPORTED NAMEQU PACKAGE_ACTIONS      *
*      SYMBOLIC PARAMETERS                                           *
*****
*      LN=(PKGAPPGR),                                                +
*      LN=(PKGBOE),                                                  +
*      LN=(PKGSHR),                                                  +
*      LN=(PKGSTAT),                                                 +
*      LN=(PKGTYPE),                                                 +
*      WARN=NO
*****
*      END OF NAMEQU SECTION                                         *
*****
NAMEQU TYPE=END
END

```

The following entries are coded in the Name Equates Table:

Entry	Comments
ESIDFLTS	This entry allows you to improve performance and establish the default behavior for all formats and traces.
FUNCEQU	This entry equates Endeavor access levels to authorization values for the RACROUTE attr= <i>auth</i> parameter.
NAMEQU	This entry creates the ENVIRONMENT_ACCESS through PACKAGE_ACTIONS pseudo data set names, which creates the entity= <i>dsname</i> value used by the RACROUTE request.

Code the ESIDFLTS, FUNCEQU and NAMEQU entries to:

- Improve performance and specify trace behaviors and diagnostic parameters (ESIDFLTS).
- Change SAF authorization levels (FUNCEQU).
- Change SAF pseudo data set name formats (NAMEQU).

The sections that follow describe how to code these entries to conform to your needs.

### 4.2.1 Defining ESI Diagnostics

The ESI Defaults macro (ESIDFLTS) allows you to specify how you manage ESI diagnostics. You can use the ESIDFLTS macro to write a diagnostic trace, to improve performance and to enable warning mode. For more information about tracing and warning mode see 4.7, “The Endeavor ESI Trace Facility” on page 4-54 and 4.6, “Using Endeavor ESI Warning Mode” on page 4-52.

The security look aside table (LAT) feature allows you to reduce the number of calls to the SAF interface and thereby improve performance. The result of each resource access request to SAF is stored in the LAT. ESI checks the LAT first for authorization and if the resource access request is listed ESI does not make a call to SAF.

**Note:** Do not use the LAT feature in an Endeavor/ROSCOE environment.

Computer Associates recommends using the default settings in the sample ESIDFLTS macro included with your installation kit. You can define only one ESIDFLTS macro in the Name Equates table and you should list the ESIDFLTS macro first. The following excerpt from a sample Name Equates Table shows the ESIDFLTS macro with the LAT feature enabled.

```

C          1      1                                7 7
1          0      6                                0 2

          ESIDFLTS TITLE='BC1TNEQU',                X
                    HEADER=ALL,                    X
                    LATSIZE=5,                     X
                    WARN=NO

```

To turn on or off the LAT feature code the ESIDFLTS entry according to the following syntax:

```

ESIDFLTS TITLE='string'
LATSIZE=n

```

Where:

- TITLE specifies a character string defined in the Name Equates Table.
- LATSIZE specifies the number of 4K pages used to store access entries in the look aside table (LAT).

The format of the ESIDFLTS macro is listed below:

**Where:**

DESC=	Specifies the descriptor codes used with the Write to Operator (WTO) messages when the trace is written to the operator's console. WTO is the default if DESC= is not specified.
HEADER=( <i>ALL/ NONE</i> )	Specifies whether to write the header information to the trace destination when opening the EN\$TRESI DD. ALL (or YES, which is synonymous) specifies that the header is written. NONE specifies that header information is not written.
LATSIZE= <i>n</i>	Specifies the number of 4K pages used to store access entries in the look aside table (LAT). There are approximately 35 entries per 4K page. Recommended number of pages is 2-10. The default, LATSIZE =0 turns off the LAT.
ROUTCDE=	Specifies the routing codes used with the Write to Operator (WTO) message when trace information is written to the operator's console. The default is WTO if ROUTCDE= is not specified.
TITLE=	Specifies a character string defined in the Name Equates Table. The title is displayed in the trace header. The default TITLE is 'No Title Specified'. The string 'BC1TNEQU' is added to the specified string.
WARN=( <i>NO/YES</i> )	Specifies warning mode for the entire table. Individual formats can override the WARN= setting. See 4.6, "Using Endeavor ESI Warning Mode" on page 4-52 for more information about warning mode.

### 4.2.2 Defining SAF Authorization Levels

When Endeavor ESI issues a RACROUTE request using ACTION\_INITIATION at the Action Initiation security control point or PACKAGE\_ACTIONS at the Package Actions security control point, it determines the SAF authorization level (*attr=auth*) by checking the FUNCEQU entry.

ENVIRONMENT\_ACCESS and PRIMARY\_OPTIONS are always issued with a READ request and cannot be modified through the Name Equates table.

## 4.2 The Name Equates Table

---

C	1	1	7 7
1	0	6	0 2

FUNCEQU	SAFAUTH=READ,	X
	C1ACTNS=(ADD,ARCHIVE,DELETE,	X
	DISPLAY,ENVRMGR,GENERATE,MOVE,	X
	PBACKOUT,PCAST,PCOMMIT,PCREATE,PDISPLAY,	X
	PEXECUTE,PLIST,PMODIFY,PREVIEW,	X
	PUTILITY,RESTORE,RETRIEVE,	X
	SIGNIN,SIGNOVR,TRANSFER,UPDATE)	
FUNCEQU	TYPE=END	

To change the mapping of access level to authorization values for ACTION\_INITIATION or PACKAGE\_ACTIONS, code the FUNCEQU entry according to the following syntax:

```
FUNCEQU SAFAUTH=(auth),  
C1ACTNS=(claccess,claccess,...,claccess)
```

---

### Where:

<i>auth</i>	The SAF authorization value equated with the Endeavor access level ( <i>claccess</i> ). Valid <i>auth</i> values follow: <ul style="list-style-type: none"><li>■ NONE</li><li>■ READ</li><li>■ UPDATE</li><li>■ CONTROL</li><li>■ ALTER</li></ul>
-------------	---

---

---

**Where:**

---

*cIaccess* The Endeavor access level equated with the SAF authorization value (*auth*). Valid *cIaccess* levels are:

- ADD
- ARCHIVE
- DELETE
- DISPLAY
- ENVRNMGR
- GENERATE
- MOVE
- PBACKOUT
- PCAST
- PCOMMIT
- PCREATE
- PDISPLAY
- PEXECUTE
- PLIST
- PMODIFY
- PREVIEW
- PUTILITY
- RETRIEVE
- SIGNIN
- SIGNOVR
- UPDATE

You can associate each *cIaccess* value with only one *auth* value. For example, you cannot associate the GENERATE *cIaccess* value to both READ and CONTROL *auth* values.

---

If you code an *auth* value of NONE, a security check (a RACROUTE request) is not issued for the *cIaccess* functions it covers. Omitting C1ACTN from the definitions results in a default assignment of NONE. Use NONE when a security check is not desired. For example, the following code results in no security with DISPLAY and RETRIEVE.

```
FUNCEQU SAFAUTH=(NONE),                                     X
      C1ACTNS=(DISPLAY,RETRIEVE)
```

#### 4.2.2.1 Mapping Authorization Values to RACF, eTrust CA-ACF2, and eTrust CA-Top Secret

eTrust CA-ACF2, and eTrust CA-Top Secret users should be aware that a secondary mapping of the RACROUTE authorization value to eTrust CA-Top Secret or eTrust CA-ACF2 equivalents occurs in the SAF interface supplied with eTrust CA-ACF2 or eTrust CA-Top Secret. These values are listed in the following table:

<b>SAF Value</b>	<b>RACF Values</b>	<b>eTrust CA-ACF2 Value</b>	<b>eTrust CA-Top Secret Value</b>
Read	Read	Read	Read
Update	Update	Write	Update
Control	Control	Write	Control
Alter	Alter	Allocate	Control

**Note:** If you specify a value other than READ you may need to update the SAF authorization when a new release of your site security package alters mapping values.

The required access level is determined at the action initiation security control point. You should be aware that eTrust CA-ACF2, and eTrust CA-Top Secret downgrade the control authority to UPDATE or WRITE for non-VSAM data sets. You must take this into account when setting up ESI security rules.

#### 4.2.2.2 FUNCEQU Authorization

The FUNCEQU example provided in Defining SAF Authorization Levels, earlier in this chapter, shows a single level authorization. This means that all MENUAUTH/ACTIONS are defined in the FUNCEQU entry with a single SAF value or attribute level of READ. When a user requests an action, the pseudo data set is built and is passed to the site security package. The user's authorization level is returned to ESI and the user can invoke the action for as long as he/she has READ access to the pseudo data set.

You can also secure actions by modifying the FUNCEQU entry. Actions should be logically grouped and mapped to different SAF values or attribute levels. When the ACTION\_INITIATION or PACKAGE\_ACTIONS security control point is encountered, the pseudo data set is built and passed to the site security package. The user's authorization level is returned to ESI and is compared to the SAF value coded for the action. If the user's authorization level is equal to or greater than the level defined in the SAFAUTH parameter the action is allowed. Otherwise, the action is denied.

An example of a modified FUNCEQU entry with four authorization levels is shown below:

---

```

C      1      1      7 7
1      0      6      0 2

FUNCEQU SAFAUTH=READ, X
      C1ACTNS=(RETRIEVE,SIGNIN,PDISPLAY,PLIST)
FUNCEQU SAFAUTH=UPDATE, X
      C1ACTNS=(ADD,UPDATE,GENERATE)
FUNCEQU SAFAUTH=CONTROL, X
      C1ACTNS=(MOVE,SIGNOVR,ARCHIVE,DELETE)
FUNCEQU SAFAUTH=ALTER, X
      C1ACTNS=(ENVRMGR, X
      PCREATE,PCAST,PREVIEW,PEXECUTE, X
      PBACKOUT,PCOMMIT,PUTILITY) X
FUNCEQU TYPE=END

```

In order to initiate a MOVE, SIGNOUT OVERRIDE, ARCHIVE, or DELETE action, the user must have control authority to the pseudo data set. Because DISPLAY is not explicitly coded in a FUNCEQU entry and therefore defaults to SAFAUTH=NONE, all users are granted DISPLAY access provided they pass other appropriate security control points.

### 4.2.3 Defining SAF Name Formats

ESI determines data set name formats by checking the NAMEQU entry in the Name Equates Table. The following excerpt from the sample Name Equates Table shows the NAMEQU entries that create the ENTITY=*dsname* value used by the RACROUTE request.

C	1	1	7 7
1	0	6	0 2
		NAMEQU ENVIRONMENT_ACCESS,	+
		L1=('C1'),	+
		L2=('ENVIRON'),	+
		L3=(ENVIRONMENT)	
		NAMEQU PRIMARY_OPTIONS,	+
		L1=('C1'),	+
		L2=(ENVIRONMENT),	+
		L3=('PMENU'),	+
		L4=(MENUITEM)	
		NAMEQU FOREGROUND_OPTIONS,	+
		L1=('C1'),	+
		L2=(ENVIRONMENT),	+
		L3=('FORACTN'),	+
		L4=(MENUITEM)	
		NAMEQU ACTION_INITIATION,	+
		L1=('C1'),	+
		L2=(ENVIRONMENT),	+
		L3=(SYSTEM),	+
		L4=(SUBSYSTEM)	
		NAMEQU ACTION_INITIATION,	+
		L1=('C1'),	+
		L2=(MENUAUTH)	
		NAMEQU PACKAGE_ACTIONS,	+
		L1=('C1'),	+
		L2=('PACKAGE'),	+
		L3=(MENUITEM),	+
		L4=(PKGSUBFC),	+
		L5=(PKGID)	

Each format (ENVIRONMENT\_ACCESS through PACKAGE\_ACTIONS) is defined only once within the Name Equates Table. (An optional second format is allowed for ACTION\_INITIATION.)

Many installations have unique naming standards and index levels for data set names. ESI allows you to customize data set names to conform to your site's conventions. In addition, you can establish your own security levels for any given field.

**Note:** These names only represent data set access rules. There are no physical data sets associated with these rules.

### 4.2.3.1 Changing Names Generated at Security Control Points

To change the names generated at each security point, code the NAMEQU entry according to the following syntax:

C	1	2	3	4	5	6	7 7
1	0	0	0	0	0	0	0 2
NAMEQU FORMAT,							X
Ln=(field1(begin,length),...fieldn(begin,length)),							X
CLASS=classname,							X
LOG=(ASIS NONE NOFAIL NOSTAT),							X
WARN=(YES NO)							
NAMEQU TYPE=END							

**Where:**

security call FORMAT	The data set name format (ENVIRONMENT_ACCESS through PACKAGE_ACTIONS).
<i>Ln</i>	The index level of the data set name. Data set names are generated in the form of: L1.L2.L3.L4. Note that the index levels generated are separated from other levels by a period (.). <i>n</i> specifies a value from 1-10. The <i>fieldn</i> values described next specify the index levels.
<i>fieldn</i> (begin, length)	A literal or an Endeavor keyword which is placed into the specified index level. <i>Fieldn</i> can be any literal of up to eight characters. Note that literals must be enclosed in single quotation marks. ( <i>begin,length</i> ) is an optional parameter which allows you to specify a portion of an Endeavor keyword. See 4.2.3.2, "Specifying a Substring of a Keyword" on page 4-16 for further information about the <i>begin</i> and <i>length</i> parameters.
<i>classname</i>	The literal DATA SET or a user-defined resource class. DATA SET is the default if <i>classname</i> is not coded. Note that a <i>classname</i> value other than DATASET may yield unpredictable results such as truncating the data set name. See 4.2.3.3, "Defining a Class Other Than Data Set with RACF" on page 4-21 for more detailed information.

**Where:**

LOG= (ASIS  NONE  NOFAIL  NOSTAT)	<p>Specifies the LOG= keyword specified on the RACROUTE macro. LOG determines whether an access attempt is recorded in your site's SMF data set and whether log messages are suppressed. The valid values are:</p> <ul style="list-style-type: none"> <li>■ ASIS — Access attempts are recorded as specified with the operating system's ADDSD and ALTDSD operator commands, or with the RDEFINE and RALTER commands for tape or DASD volumes (if the CLASS= parameter specifies something other than data set).</li> <li>■ NONE — The access attempt is not recorded, regardless of success or failure. <i>eTrust CA-ACF2 users should set this parameter.</i></li> <li>■ NOFAIL — The access attempt is not recorded if the authorization check fails. The access attempt is recorded if the authorization check succeeds.</li> <li>■ NOSTAT — The access attempt is not recorded and resource statistics are not updated.</li> </ul>
WARN= (YES NO)	<p>Specifies the local warning option. This parameter overrides the ESIDFLTS value. The YES option turns on warning mode for the resource name (the security control point). The NO option turns off warning mode. If YES or NO is not specified, then the action defaults to the coding specified in the ESIDFLTS WARN option. If the ESIDFLTS WARN option is not specified, the default value is WARN=NO. See 4.6, “Using Endeavor ESI Warning Mode” on page 4-52.</p>
TYPE=END	<p>Indicates the end of the name equates entries and can be specified separately or on the last NAMEQU FORMAT<math>n</math> entry.</p>

**4.2.3.2 Specifying a Substring of a Keyword**

You can specify a substring of a keyword, according to the following syntax:

$L_n = (\text{KEYWORD}(B, L))$

**Where:**

B	The first character of the keyword (beginning column relative to one).
L	The number of characters to extract from the keyword.

For example, to obtain a field value of FIN to represent a system named FINANCE, you would code the following:

L1=(SYSTEM(1,3))

You can concatenate field values by specifying more than one field, each separated by a comma. For example, to obtain a field value of ENVIPROD to represent an environment named PRODUCTION, you would code the following:

L2=('ENVI',ENVIRONMENT(1,4))

**Note:** When data set names are generated, all trailing and embedded blanks are compressed. A value of '\$' occurs in the index level of the name if the resulting index level is all blanks (that is, not available).

The following table lists the Endeavor keywords and the formats for which the keywords are available.

Endeavor Keyword	ENV	PRI	FOR	ACTS	PKG
ENVIRONMENT	X	X	X	X	
ACTION		X	X	X	X
MENUITEM		X	X	X	X
MENUAUTH		X	X	X	X
SYSTEM				X	
SUBSYSTEM				X	
STAGEID				X	
STAGENAME				X	
STAGENO				X	
TYPE				X	
ELEMENT				X	
ELM-10				X	
CCID				X	
PKGSUBFC					X
PKGID					X
PKGTYPE					X
PKGSTAT					X
PKGAPPGR					X
PKGBOE					X

Endevor Keyword	ENV	PRI	FOR	ACTS	PKG
PKGSHR					X

**Note:** The format names used in the table have been abbreviated as: ENVIRONMENT\_ACCESS=ENV, PRIMARY\_OPTIONS=PRI, FOREGROUND\_OPTIONS=FOR, ACTION\_INITIATION=ACTS, and PACKAGE\_ACTIONS=PKG.

A brief definition of each keyword follows:

Keyword	Definitions
ENVIRONMENT <b>1</b>	The Endevor environment name
ACTION	The Endevor access level for which a request is made. The ACTION keyword is interchangeable with the MENUAUTH keyword. The following values are valid for action: ADD, ARCHIVE, DELETE, DISPLAY, ENVRNMGR, GENERATE, MOVE, PBACKOUT, PCAST, PCOMMIT, PCREATE, PDELETE, PDISPLAY, PEXECUTE, PEXPORT, PLIST, PMODIFY, PRESET, PREVIEW, PUTILITY, RETRIEVE, SIGNIN, SIGNOVR, UPDATE.

<b>Keyword</b>	<b>Definitions</b>
MENUIITEM	<p>Allows individual line tailoring of the Primary Options menu, the Foreground Options menu, and the Package Actions menu.</p> <p>When used with PRIMARY_OPTIONS (Primary Options menu), the following values are valid: DISPLAY, FOREGRND, BATCH, PACKAGE, BATCHPKG, USER, ENVRMENT.</p> <p>When used with FOREGROUND_OPTIONS (Foreground Options menu), the following values are valid: DISPLAY, ADDUPDT, RETRIEVE, GENERATE, MOVE, DELETE, PRINT, SIGNIN.</p> <p>When used with ACTION_INITIATION, the following values are valid: ADD, UPDATE, DISPLAY, RETRIEVE, GENERATE MOVE, DELETE, PRINT, RESTORE, SIGNIN, TRANSFER, LIST, ARCHIVE.</p> <p>When used with PACKAGE_ACTIONS, the following values are valid: BACKOUT, CAST, COMMIT, CREATE, DISPLAY, EXECUTE, MODIFY, REVIEW, UTILITY.</p> <p>See the descriptions of:</p> <ul style="list-style-type: none"> <li>■ 4.2.4.2, “PRIMARY_OPTIONS — Primary Options Security Control Point”</li> <li>■ -- Heading 'C43405' unknown --</li> <li>■ 4.2.4.3, “ACTION_INITIATION — Action Initiation Security Control Point (Standard)”</li> <li>■ 4.2.4.4, “ACTION_INITIATION — Action Initiation Security Control Point (Extension)”</li> </ul> <p>for more information.</p>
MENUAUTH	<p>The Endeavor authorization level for which a request is made. The MENUAUTH keyword is interchangeable with the ACTION keyword. The following values are valid for MENUAUTH: ADD, ARCHIVE, DELETE, DISPLAY, ENVRNMGR, GENERATE, MOVE, PBACKOUT, PCAST, PCOMMIT, PCREATE, PDISPLAY, PEXECUTE, PLIST, PMODIFY, PREVIEW, PUTILITY, RETRIEVE, SIGNIN, SIGNOVR, UPDATE.</p>
SYSTEM <b>1</b>	The Endeavor system name.
SUBSYSTEM <b>1</b>	The Endeavor subsystem name.
STAGEID	The Endeavor stage ID (1 character value) is taken from the STG1= and STG2= parameters in the C1DEFLT5 Table. The values come from the C1DEFLT5 TYPE=ENVIRONMENT.

<b>Keyword</b>	<b>Definitions</b>
STAGENAME <b>1</b>	The Endeavor stage name.
TYPE <b>1</b>	The Endeavor element type.
ELEMENT	The first eight characters of the Endeavor element name.
ELEM-10	The 10 characters of the Endeavor element name.
CCID	The current Change Control Identifier. A CCID can be up to 12 characters.
PKGSUBFC <b>2</b>	The sub-menu function code. The following values are valid: <ul style="list-style-type: none"> <li>▪ ACTSUMM</li> <li>▪ APPROVE</li> <li>▪ APPROVER</li> <li>▪ BACKIN</li> <li>▪ BACKOUT</li> <li>▪ BUILD</li> <li>▪ CAST</li> <li>▪ COMMIT</li> <li>▪ COPY</li> <li>▪ DELETE</li> <li>▪ DENY</li> <li>▪ DISPLAY</li> <li>▪ EDIT</li> <li>▪ EXPORT</li> <li>▪ IMPORT</li> <li>▪ LIST</li> <li>▪ REPORTS</li> <li>▪ RESET</li> <li>▪ SCL</li> </ul>
PKGID	The user-defined package name.
PKGTYPE <b>3</b>	Defines whether the package is emergency or standard: <ul style="list-style-type: none"> <li>▪ STANDARD</li> <li>▪ EMERGENCY</li> </ul>
PKGSTAT <b>4</b>	The package status: <ul style="list-style-type: none"> <li>▪ IN-EDIT</li> <li>▪ IN-APPROVAL</li> <li>▪ DENIED</li> <li>▪ APPROVED</li> <li>▪ IN-EXECUTION</li> <li>▪ EXECUTED</li> <li>▪ COMMITTED</li> </ul>
PKGAPPGR	The approver group name.

<b>Keyword</b>	<b>Definitions</b>
PKGBOE	The backout-enabled status of the package: <ul style="list-style-type: none"> <li>▪ Y</li> <li>▪ N</li> </ul>
PKGSHR	The share option associated with the package: <ul style="list-style-type: none"> <li>▪ Y</li> <li>▪ N</li> </ul>

**Note:** For the package symbolics, a value of '\$' is substituted for the variable when the real value is not available to Endeavor. For example, prior to a CAST, the value for PKGAPPGR is a \$.

**1**: Up to eight characters in length.

**2**: The value for PKGSUBFC always resolves to LIST when selecting an option from the package Foreground Options menu. The values listed above are only available from the corresponding foreground panel. For example, PKGSUBFC is resolved to LIST when selecting option 3 from the package Foreground Options menu and CAST from the Cast panel.

**3**: Each node in a data set name only allows a maximum of eight characters per value; therefore, values greater than eight characters are truncated to eight characters (e.g., EMERGENC).

**4**: PKGSTAT status names can be more than eight characters long (OS/390 only supports eight characters in a data set name node). Use substrings to limit the number of characters in the generated node. By default, the value is the first eight characters of the status name (e.g., IN-APPROVAL truncates to IN-APPRO).

**CAUTION:**

**If the pseudo data set name is greater than 44 characters, it is truncated.**

### 4.2.3.3 Defining a Class Other Than Data Set with RACF

The RACROUTE macro has a default value of DATASET for the CLASS parameter. Depending on the granularity of your site security rules you could have many rules that belong in this category. You can reduce the number of rules that are searched during authorization by creating an additional class category that is specific to ESI security rules.

The following procedure explains how to define a class other than DATASET if you have installed RACF as your site security package. In this example the resource class is called \$ENDEVOR.

1. Add the resource class name \$ENDEVOR to the RACF class description table (CDT). See the *RACF Macros Interface Manual* for information about parameters. Use the following macro to create a class called \$ENDEVOR.

C	1	2	3	4	5	6	7	7
1	0	0	0	0	0	0	0	2

```

$ENDEVOR ICHERCDE CLASS=$ENDEVOR,
ID=141,
MAXLNTH=246,
FIRST=ALPHANUM,
OTHER=ANY,
POSIT=20,
RACLIST=ALLOWED,
DFTUACC=NONE,
OPER=NO

```

```

ICHRRUDE ICHERCDE
END

```

```

/*
/**
//INSTL3 EXEC IPOSMPF, COND=(0,NE),
//          CSI='MVSSMPF.GLOBAL.CSI'
//SMPEIN DD *
SET BDY(GLOBAL).
REJECT S(NRACCDT) BYPASS(APPLYCHECK).
RECEIVE S(NRACCDT) SYSMODS.
SET BDY(M220TAF).

```

**Note:** ESI supports resource names up to 246 bytes long. In order for you to use extended resource names, an installation-defined Class must be used. In addition, you must insure that your site's security package supports extended names. Refer to the documentation for your security package.

2. Add the entry in the following example to the RACF router table. The processor below executes these two steps.

```

TAB16 ICHRFRTB CLASS=$ENDEVOR,ACTION=RACF
TABEND ICHRFRTB TYPE=END
END ICHRFR01

```

```

/*
/**
//INSTL3 EXEC IPOSMPF,COND=(4,NE),
//          CSI='MVSSMPF.GLOBAL.,CSI'
//SMPEIN DD *
SET BDY(GLOBAL) B .
REJECT S(NRACRTT) BYPASS(APPLYCHECK) .
RECEIVE S(NRACRTT) SSMODS .
SET DBY (M220TAF) .
APPLY S(NRACRTT) REDO
/*
//SMPPTFIN DD DSN=&&LOADSET,DISP=(OLD,DELETE)

```

3. Assemble the CDT and the router table and IPL the system.
4. Activate the \$ENDEVOR resource class using the RACF SETROPTS command. This activates generic access checking for the resource class.
5. Modify the Name Equates Table. The following table show the changes that should be made to the NAMEQU entries.

C	1	2	3	4	5	6	7 7
1	0	0	0	0	0	0	0 2
NAMEQU	ENVIRONMENT_ACCESS,						X
	L1=('C1'),						X
	L2=('ENVIRON'),						X
	L3=(ENVIRONMENT),						X
	CLASS='\$ENDEVOR'						
NAMEQU	PRIMARY_OPTIONS,						X
	L1=('C1'),						X
	L2=('PMENU'),						X
	L3=(ENVIRONMENT),						X
	L4=(MENUITEM),						X
	CLASS='\$ENDEVOR'						
NAMEQU	BACKGROUND_OPTIONS,						X
	L1=('C1'),						X
	L2=('FMENU'),						X
	L3=(ENVIRONMENT),						X
	L4=(MENUITEM),						X
	CLASS='\$ENDEVOR'						
NAMEQU	ACTION_INITIATION,						X
	L1=('C1'),						X
	L2=(ENVIRONMENT),						X
	L3=(SYSTEM),						X
	L4=(SUBSYSTEM),						X
	L5=(MENUAUTH),						X
	CLASS='\$ENDEVOR'						
NAMEQU	PACKAGE_ACTIONS,						X
	L1=('C1'),						X
	L2=('PACKAGE'),						X
	L3=(MENUITEM),						X
	L4=(PKGSUBFC),						X
	L5=(PKGID),						X
	CLASS='\$ENDEVOR'						

#### 4.2.4 Security Rule Formats ENVIRONMENT\_ACCESS Through PACKAGE\_ACTIONS

The RACROUTE request is derived from rules you define in the Name Equates Table. Formats ENVIRONMENT\_ACCESS through ACTION\_INITIATION coordinate access levels, menu options, and authorization levels with the site security packages (RACF, eTrust CA-ACF2, and eTrust CA-Top Secret).

Formats are defined in the Name Equates Table, a portion of which is shown below.

C	1	1					7 7
1	0	6					0 2
NAMEQU	ENVIRONMENT_ACCESS,						+
	L1=('C1'),						+
	L2=('ENVIRON'),						+
	L3=(ENVIRONMENT)						

---

```

NAMEQU PRIMARY_OPTIONS,
    L1=('C1'),
    L2=(ENVIRONMENT),
    L3=('PMENU'),
    L4=(MENUITEM)
NAMEQU FOREGROUND_OPTIONS,
    L1=('C1'),
    L2=(ENVIRONMENT),
    L3=('FORACTN'),
    L4=(MENUITEM)
NAMEQU ACTION_INITIATION,
    L1=('C1'),
    L2=(ENVIRONMENT),
    L3=(SYSTEM),
    L4=(SUBSYSTEM)
NAMEQU ACTION_INITIATION,
    L1=('C1'),
    L2=(MENUAUTH)
NAMEQU PACKAGE_ACTIONS,
    L1=('C1'),
    L2=('PACKAGE'),
    L3=(MENUITEM),
    L4=(PKGSUBFC),
    L5=(PKGID)

```

Under each format entry, rules define parts of every RACROUTE request. The format defines how the pseudo data set is built. Variables in each rule appear without single quotes and literals appear within single quotes. Endeavor substitutes the appropriate value for each variable.

1. Environments (**ENVIRONMENT\_ACCESS**): Restricts user access to environments. ENVIRONMENT\_ACCESS calls are issued during Endeavor initialization both in foreground and batch.
2. Primary Options panel (**PRIMARY\_OPTIONS**): The primary options panel is customized for each user, based on the rules set up by the security administrator. Only options that the user can select are displayed on the Primary Options panel. PRIMARY\_OPTIONS calls are issued before the Primary Options panel is displayed.
3. Foreground Options panel (**FOREGROUND\_OPTIONS**): Only options that the user can select are displayed on the Foreground Options panel. FOREGROUND\_OPTIONS calls are issued before the Foreground Options panel is displayed.
4. Action initiation (**ACTION\_INITIATION**): Prior to performing a selected action, ESI requests a ruling to determine whether the user has the authorization to perform the action. ESI makes a pass for each defined ACTION\_INITIATION format.
5. Action initiation (**ACTION\_INITIATION**): The extension ACTION\_INITIATION is an optional extension of the standard ACTION\_INITIATION to allow for names longer than 44 bytes. This format is called only if the standard ACTION\_INITIATION is successful.

6. Package actions (**PACKAGE\_ACTIONS**): Prior to performing an action against a package, ESI requests a ruling to determine whether the user has the authorization to perform that action against the package.

#### 4.2.4.1 ENVIRONMENT\_ACCESS—Environment Access Security Control Point

The Environment access security control point occurs at the following places:

- During Endeavor initialization, prior to display of the Environment Selection menu in the foreground and prior to processing of any actions in batch.
- During UNLOAD and RELOAD operations to re-verify the user's authority to backup and restore the desired inventory locations.

**Note:** When the user changes environments in the foreground, Endeavor looks up in the user's accessible environments to see whether the user has access to the environment. No ESI call is issued during this processing because it is not necessary.

Use the Environment Selection menu to select an environment. Any other panel that displays the ENVIRONMENT field allows the user to switch environments.

When Endeavor starts under ISPF, Endeavor ESI issues a RACROUTE request using ENVIRONMENT\_ACCESS for each environment defined in the C1DEFLT5 Table. Every authorized environment is then displayed on the user's Environment Selection menu.

Define rules for your site security package based on the ENVIRONMENT\_ACCESS data set names. These rules determine which environment(s) are accessible to the user and are displayed on the Environment Selection menu. A user must have READ authority for each data set name to gain access to the specified environment.

**Sample:** A sample of ENVIRONMENT\_ACCESS rules is shown below.

```

C          1      1                      7 7
1          0      6                      0 2
NAMEQU ENVIRONMENT_ACCESS,
          L1=('C1'),
          L2=('ENVIRON'),
          L3=(ENVIRONMENT)

```

This becomes: C1.ENVIRON.*environment*

Where *environment* is the name for which access is requested. If the user has access to only one environment, the Environment Selection menu is not presented. If the user does not have access to any environments, Endeavor is not available.

**Note:** If you use environment mapping you must also have access to all forward environments up the map.

**Example:** The security administrator at a site with two environments (QA and PROD) wants to give a programmer access to both environments. To do this he/she must define a data set access rule for the site security package that gives the programmer READ access to the data set names:

C1.ENVIRON.QA  
C1.ENVIRON.PROD

#### 4.2.4.2 PRIMARY\_OPTIONS — Primary Options Security Control Point

The Primary Options security control point occurs prior to building the Primary Options menu for the current environment in foreground, after access is granted through the Environment Selection menu or during UNLOAD/RELOAD.

Define rules for your site security package based on the PRIMARY\_OPTIONS data set names to determine the option(s) to be displayed on the user's Primary Options menu. A user must have READ authority for each data set name to gain access to the specified primary option.

*Sample:* A sample of PRIMARY\_OPTIONS rules is shown below.

```
C          1      1                               7 7
1          0      6                               0 2

NAMEQU  FOREGROUND_OPTIONS,
        L1=(^C1'),
        L2=(ENVIRONMENT),
        L3=('FORACTN'),
        L4=(MENUITEM)
```

This becomes: *C1.environment.FORACTN.menuitem*

---

#### Where:

---

<i>environment</i>	The environment the user is trying to access.
<i>menuitem</i>	The corresponding List Menu Item value, as shown below in the following table.

---

Value	Item Displayed
DISPLAY	DISPLAY
ADDUPDT	ADD/UPDATE
RETRIEVE	RETRIEVE
GENERATE	GENERATE
MOVE	MOVE
DELETE	DELETE
PRINT	PRINT
SIGNIN	SIGNIN

**Note:** This security point can only be reached if a PRIMARY\_OPTIONS rule allows access to the Foreground Actions menu for the current environment.

BACKGROUND\_OPTIONS rules do not apply to batch operations.

**Example:** The security administrator at a site with an environment called QA wants to give a programmer access to the ADD/UPDATE, RETRIEVE, PRINT, and SIGNIN options. To do this, he/she must define a data set access rule for the site security package (RACF, eTrust CA-ACF2, and eTrust CA-Top Secret) that gives the programmer READ access to the data sets:

```
C1.QA.FORACTN.ADDUPDT
C1.QA.FORACTN.RETRIEVE
C1.QA.FORACTN.PRINT
C1.QA.FORACTN.SIGNIN
```

#### 4.2.4.3 ACTION\_INITIATION — Action Initiation Security Control Point (Standard)

The Action Initiation security control point occurs:

- Prior to performing an Endeavor action.
- Prior to a cast operation during package processing, for each action in a package, if the PKGCSEC flag is set to Y.
- During package verification processing.

Define rules for the site security package based on the ACTION\_INITIATION data set names. These rules determine the Endeavor actions the user can perform. A user must have the proper level of authority to each data set name (based on action) to gain access to the specified Endeavor action.

You need to write rules to secure the source and target locations for ACTION\_INITIATION.

*Sample:* Sample ACTION\_INITIATION rules are shown below.

```
C      1   1   2       3       4       5       6       7 7
1      0   6   0       0       0       0       0       0 2
```

```
NAMEQU ACTION_INITIATION,
        L1=('C1'),
        L2=(ENVIRONMENT),
        L3=(SYSTEM),
        L4=(SUBSYSTEM)
```

This becomes: *C1.environment.system.subsystem*

---

#### Where:

---

*environment*      The environment the user is trying to access.

---

*system*            The system the user is trying to access.

---

*subsystem*        The subsystem the user is trying to access.

---

See the example described in “Example” on page 4-28.

#### 4.2.4.4 ACTION\_INITIATION — Action Initiation Security Control Point (Extension)

The extension ACTION\_INITIATION is an optional extension of the standard ACTION\_INITIATION that can allow for names longer than the class attribute allows. Access to both ACTION\_INITIATIONS are required and you must write rules to secure the source and target locations for them.

**Note:** This is only called if ACTION\_INITIATION has RC=0.

*Sample:* Sample extension ACTION\_INITIATION rules are shown below.

```
C      1      1      7 7
1      0      6      0 2
```

```
NAMEQU ACTION_INITIATION,
        L1=('C1'),
        L2=(MENUAUTH)
```

This becomes: *C1.menuauth*

Where *menuauth* is the access level required for the requested action.

**Note:** The table in 4.2.4.5, “The Default Authorization Value” provides translated default authorization values for each RACROUTE request.

**Example:** This example demonstrates how both ACTION\_INITIATIONS rules work together. Assume you are a security administrator and you need to define data set access rules for your site security package. You have an environment called QA and you want to give a programmer access to a system called FINANCE and a subsystem called ACTSPAY. In addition, you want to limit the actions the programmer can perform in the subsystem ACTSPAY to RETRIEVE and DISPLAY. To accomplish this you must write both ACTION\_INITIATIONS rules. These rules grant the programmer READ access to these pseudo data sets:

```
C1.QA.FINANCE.ACTSPAY (ACTION_INITIATION)
C1.RETRIEVE (ACTION_INITIATION)
```

#### 4.2.4.5 The Default Authorization Value

The default *authorization* value for each RACROUTE request is translated in the following table. The table reflects the sample BC1TNEQU table included with your installation kit.

To Perform This Activity	Access Level Required	SAF Authorization Level
Add	Add	READ
Update	Update	READ
Retrieve	Retrieve	READ

<b>To Perform This Activity</b>	<b>Access Level Required</b>	<b>SAF Authorization Level</b>
Generate (Stage 1)	Generate	READ
Generate (Stage 2)	Move	READ
Move (from Stage 1)	Move	READ
Move (from Stage 2)	Move	READ
Move (to Stage 1)	Add	READ
Move (to Stage 2)	Move	READ
Display from selection list	Display	READ
Browse	Retrieve	READ
Delete (Stage 1)	Delete	READ
Delete (Stage 2)	Move	READ
Signin	Signin	READ
Print	Display	READ
Transfer (to Stage 1)	Add/Update	READ
Transfer (to Stage 2)	Move	READ
Transfer (from Stage 1 with Delete)	Delete	READ
Transfer (from Stage 2 with Delete)	Move	READ
Transfer (from Stage 1 without Delete)	Retrieve	READ
Transfer (from Stage 2 without Delete)	Retrieve	READ
Archive	Archive	READ
Restore to Stage 1	Add	READ
Restore to Stage 2	Move	READ
Copy	None	READ
List	Display	READ
Signout Override <b>1</b>	Signovr	READ

To Perform This Activity	Access Level Required	SAF Authorization Level
Actions against any elements of type processor <b>2</b>	Envrnmgr	READ

**Note:**

**1**: Signout override is always the second call in an action. The first call is the specific action involved - Add or Delete, for example - and then, if necessary, the call for signout override is performed.

**2**: When performing actions against type processors, two calls are issued. The first call checks whether the user can perform the specific action involved, such as Add or Delete. If the user is able to perform that action, then a second call is issued for access level ENVRNMGR to see if the user also has permission to work with type processors.

When you determine the authorization access for a user be sure to check the "Access level required" column to ensure that you are not giving users access to activities that you do not want them to have access.

#### 4.2.4.6 PACKAGE\_ACTIONS — Package Actions Security Control Point

The Package Actions security control point occurs prior to performing a package action.

Define rules for the site security package based on the PACKAGE\_ACTIONS data set names. These rules determine the Endeavor actions the user can perform. A user must have the proper level of authority to each data set name (based on action) to gain access to the specified Endeavor action.

You need to write rules to secure the source and target locations for PACKAGE\_ACTIONS.

*Sample:* Sample PACKAGE\_ACTIONS rules are shown below.

```

C          1      1
1          0      6

```

7 7  
0 2

```

NAMEQU PACKAGE_ACTIONS,
        L1=('C1'),
        L2=('PACKAGE'),
        L3=(MENUITEM),
        L4=(PKGSUBFC),
        L5=(PKGID)

```

This becomes: C1.PACKAGE.menuitem.pkgsubfc.pkgid

---

**Where:**

---

*menuitem* Allows individual line tailoring of the Primary Options menu. The following values are valid:

- BACKOUT
- CAST
- COMMIT
- CREATE
- DISPLAY
- EXECUTE
- MODIFY
- REVIEW
- UTILITY

*pkgsubfc* **1** The sub-menu function code or action (e.g., CREATE allows you to build, import, export, etc.). One of the following values:

- ACTSUMM
- APPROVE
- APPROVER
- BACKIN
- BACKOUT
- BUILD
- CAST
- COMMIT
- COPY
- DELETE
- DENY
- DISPLAY
- EDIT
- EXPORT
- IMPORT
- LIST
- REPORTS
- RESET
- SCL

*pkgid* The user-defined package name

---

**Note:**

**1**: The value for *pkgsubfc* always resolves to LIST when selecting an option from the package Foreground Options menu. The values listed above are only available from the corresponding foreground panel. For example, *menuitem* is resolved to LIST when selecting option 3 from the package Foreground Options menu and CAST from the Cast panel.

---

The NAMEQU macro supports the FORMAT=PACKAGE\_ACTION entry along with the new symbolics required to build the model name. The format of the model name is variable, but can include the following specific package symbolics:

Variable	Description
PKGSHR	The share option associated with the package: <ul style="list-style-type: none"> <li>▪ Y</li> <li>▪ N</li> </ul>
PKGBOE	The backout-enabled status of the package: <ul style="list-style-type: none"> <li>▪ Y</li> <li>▪ N</li> </ul>
PKGAPPGR	Approver group name
PKGSTAT <b>1</b>	The package status: <ul style="list-style-type: none"> <li>▪ IN-EDIT</li> <li>▪ IN-APPROVAL</li> <li>▪ DENIED</li> <li>▪ APPROVED</li> <li>▪ IN-EXECUTION</li> <li>▪ EXECUTED</li> <li>▪ COMMITTED</li> </ul>
PKGTYPE <b>2</b>	Defines whether the package is emergency or standard: <ul style="list-style-type: none"> <li>▪ STANDARD</li> <li>▪ EMERGNCY</li> </ul>

**Note:**

**1**: PKGSTAT status names can be more than eight characters long (OS/390 only supports eight characters in a data set name node). Use substringing to limit the number of characters in the generated node. By default, the value is the first eight characters of the status name (e.g., IN-APPROVAL truncates to IN-APPRO).

**2**: Each node in a data set name only allows a maximum of eight characters per value; therefore, values greater than eight characters are truncated to eight characters (e.g., EMERGENC).

#### 4.2.4.7 PACKAGE\_ACTIONS ESI Calls

The following table displays a couple of the rules that are built for this sample. The pseudo data set names varies depending on how your PACKAGE\_ACTIONS NAMEQU is configured.

Menuitem/ Subfunction	FUNCEQU Name	Sample Pseudo Data Set Security Rule <i>C1.PACKAGE.menuitem.pkgsubfc.pkgid</i>
<b>BACKOUT</b>	PLIST	C1.PACKAGE.DISPLAY.LIST.PKG001
Backout	PBACKOUT	C1.PACKAGE.BACKOUT.BACKOUT.PKG001
Display	PDISPLAY	C1.PACKAGE.DISPLAY.BACKOUT.PKG001
Backin	PBACKOUT	C1.PACKAGE.BACKOUT.BACKIN.PKG001

---

<b>Menuitem/ Subfunction</b>	<b>FUNCEQU Name</b>	<b>Sample Pseudo Data Set Security Rule C1.PACKAGE.menuitem.pkgsubfc.pkgid</b>
<b>CAST</b>	PLIST	C1.PACKAGE.DISPLAY.LIST.PKG001
Cast	PCAST	C1.PACKAGE.CAST.CAST.PKG001
SCL	PDISPLAY	C1.PACKAGE.DISPLAY.SCL.PKG001

---

**Note:** There is no security for the Notes subfunction.

---

## 4.3 Enabling Endeavor ESI

This section explains how to enable Endeavor ESI, which is supplied on the Endeavor installation tape. The procedure for enabling Endeavor ESI should be performed after the Endeavor product is installed and verified.

To enable Endeavor ESI, follow these steps:

1. Prepare your security worksheet.
2. Define rules for your site security package.
3. Customize the Name Equates Table (Formats ENVIRONMENT\_ACCESS through PACKAGE\_ACTIONS).
4. Assemble and link-edit the Name Equates Table (BC1TNEQU).
5. Assemble and link the Endeavor Defaults Table (C1DEFLTS).
6. Test ESI security using warning mode.

The sections that follow explain each of these steps in greater detail.

### 4.3.1 Preparing Your ESI Security Worksheet

To begin planning your ESI security strategy you must gather information about all the users and applications for which you are responsible. You need to determine how users and applications relate to each other and how detailed you want your security strategy to be.

Computer Associates recommends using an ESI worksheet to help you define and categorize the various levels of access required by anyone involved in application development. Once you have completed your ESI worksheet, you can begin defining access levels.

**Note:** Blank worksheets for defining security rules are provided in Appendix A, “Security Worksheets.”

Keep an updated version of your ESI worksheet available for reference purposes. If you make changes to your Names Equates table or your site security package rules, remember to update your ESI worksheet as well. Use the worksheet to:

- Associate authorization levels with personnel or departments.
- List activities to include for selected departments in Stages 1 and 2 of each environment.
- Refer to when updating or revising your security rules.

## 4.4 Defining Security Rules for Your Site

The following section contains four sample procedures that describe the steps you need to follow in order to determine security rules for your site. The procedures describe steps for defining the following security rules:

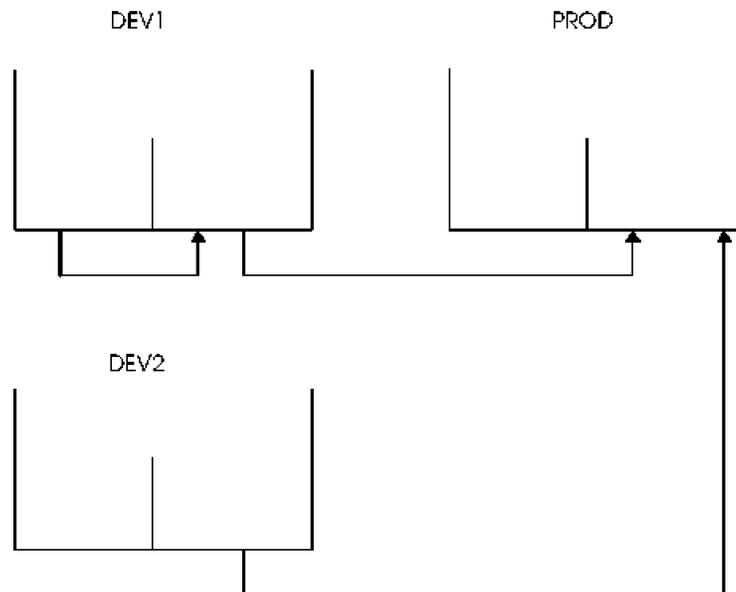
- Site environment rules
- Primary Options panel rules
- Foreground action rules
- Action initiation rules
- Package action rules

**Note:** Blank worksheets for defining security rules are provided in Appendix A, “Security Worksheets.”

### 4.4.1 Defining Security Rules for Your Site Environments

The following four steps explain how to set up security for the environments at your site.

1. Plan a diagram of the flow for your site environments and label each box with the name of your environments. The following example shows a site where parallel development occurs between environments DEV1 and DEV2.



2. Plan a table that lists your environments in the far left column and the users at your site along the top row. In the following example an X indicates that a user has security access.

Envr	Appl Pgmr	Appl Mgr	Other Depts	QA	Prod Ctl	Tech Supp	Audit	Admin
DEV1	X	X				X		X
DEV2	X	X				X		X
PROD	X	X	X	X	X	X	X	X

3. Determine the format for your environment security rules. In order to generate a pseudo data set name in the format 'C1'.ENVIRON'.*environment*, you must set up the NAMEQU entry in the Name Equates Table using this format:

```
NAMEQU ENVIRONMENT_ACCESS,
      L1=('C1'),
      L2=('ENVIRON'),
      L3=(environment)
```

4. In the tables below, determine the pseudo data set names based on the format described in Step 3 and the environments entered in Step 2.

Env	AP	AM	OD	QA	PC	TS	AU	AD	Data Set Names
DEV1	X	X				X		X	C1.ENVIRON.DEV1
DEV2	X	X				X		X	C1.ENVIRON.DEV2
PROD	X	X	X	X	X	X	X	X	C1.ENVIRON.PROD

**Note:** User access to data sets is indicated with an X. The user must have read access to the data sets in order to have environment access.

See 4.4.1.1, “User Abbreviations” for a description of column headings.

#### 4.4.1.1 User Abbreviations

These abbreviations are used in the security table's column heading to describe the users:

- AP — Application Programming
- AM — Application Management
- OD — Other Departments
- QA — Quality Assurance
- PC — Production Control
- TS — Technical Support
- AU — Audit
- AD — Administration

## 4.4.2 Defining Security Rules for the Primary Options Panel

The following three steps explain how to set up security for the Primary Options Panel at your site.

1. Plan a table that lists the Primary Options panel menu items in the far left column and the users at your site along the top row. In the following example, an X indicates that a user has access to a menu item.

Menu Item	Appl Pgmr	Appl Mgr	Other Depts	QA	Prod Ctl	Tech Supp	Audit	Admin
DISPLAY	X	X	X	X	X	X		X
BACKGROUND	X	X				X		X
BATCH	X	X				X		X
PACKAGE				X	X	X		X
USER MENU				X	X	X		X
ENVIRONMENT						X		X
UNLOAD <b>1</b>								X
RELOAD <b>1</b>								X
BATCH <b>1</b> PACKAGE	X	X		X	X	X		X

**Note:**

**1** : Batch actions only

2. Determine the format for your Primary Options panel security rules. In order to generate a pseudo data set name in the format 'C1'.environment.'PMENU'.menuitem, you must set up the NAMEQU entry in the Name Equates Table using this format:

```
C      1      1      7 7
1      0      6      0 2
1      0      6      0      0      0      0      0      0 2
```

```
NAMEQU FOREGROUND_OPTIONS,
        L1=('C1'),
        L2=(environment),
        L3=('FORACTN'),
        L4=(menuitem)
```

3. In the tables below, determine the pseudo data set names based on the format described in Step 2 and the menu items indicated with an X in Step 1.

Menu Item	AP	AM	OD	QA	PC	TS	AU	AD	Data Set Names
DISPLAY	X	X	X	X	X	X		X	C1.DEV1.FORACTN.DISPLAY
ADD/ UPDATE								X	C1.DEV1.FORACTN.ADDUPDT
RETRIEVE	X	X				X		X	C1.DEV1.FORACTN.RETRIEVE
GENERATE								X	C1.DEV1.FORACTN.GENERATE
MOVE								X	C1.DEV1.FORACTN.MOVE
DELETE					X	X		X	C1.DEV1.FORACTN.DELETE
SIGNIN	X	X		X	X	X		X	C1.DEV1.FORACTN.SIGNIN
PRINT	X	X		X	X	X		X	C1.DEV1.FOREACTN.PRINT

**Note:** User access to data sets is indicated with an X. The user must have READ access to the data sets in order to access to the Foreground Options panel actions.

See 4.4.1.1, “User Abbreviations” for a description of column headings.

### 4.4.3 Defining Security Rules for Action Initiations

The following three steps explain how to set up security for action initiations at your site.

1. Design a table that lists the action initiation items in the far left column and the users at your site along the top row. In the following example, an X indicates that a user has access to the action.

Action	Appl Pgmr	Appl Mgr	Other Depts	QA	Prod Ctl	Tech Supp	Audit	Admin
DISPLAY	X	X	X	X	X	X		X
ADD	X	X				X		X
UPDATE	X	X				X		X
RETRIEVE	X	X	X			X		X
GENERATE	X	X				X		X
MOVE				X	X	X		X
DELETE	X	X		X	X	X		X
SIGNIN	X	X	X			X		X
SIGNOUT/ OVERRIDE		X				X		X
ARCHIVE						X		X

Action	Appl Pgrmr	Appl Mgr	Other Depts	QA	Prod Ctl	Tech Supp	Audit	Admin
RESTORE						X		X
PROCESSORS				X	X			X

2. Determine the format for your action initiation security rules. In order to generate a pseudo data set name in the format 'C1'.environment.system.subsystem.menuauth, you must set up the NAMEQU entry in the Name Equates Table using this format:

```
C      1      1      7 7
1      0      6      0 2
```

```
NAMEQU ACTION_INITIATION,
        L1=('C1'),
        L2=(environment),
        L3=(system),
        L4=(subsystem),
        L5=(menuauth)
```

3. In the tables below, determine the pseudo data set names based on the format described in Step 2 and the actions indicated with an X in Step 1.

Action	AP	AM	OD	QA	PC	TS	AU	AD	Data Set Names
DISPLAY	X	X	X	X	X	X		X	C1.DEV1.FINANCE.*.DISPLAY
ADD	X	X				X		X	C1.DEV1.FINANCE.*.ADD
UPDATE	X	X				X		X	C1.DEV1.FINANCE.*.UPDATE
RETRIEVE	X	X	X			X		X	C1.DEV1.FINANCE.*.RETRIEVE
GENERATE	X	X				X		X	C1.DEV1.FINANCE.*.GENERATE
MOVE				X	X	X		X	C1.DEV1.FINANCE.*.MOVE
DELETE	X	X		X	X	X		X	C1.DEV1.FINANCE.*.DELETE
SIGNIN	X	X	X			X			C1.DEV1.FINANCE.*.SIGNIN
SIGNOUT/ OVERRIDE		X		X		X		X	C1.DEV1.FINANCE.*.SIGNOVR
ARCHIVE						X		X	C1.DEV1.FINANCE.*.ARCHIVE
PROCESSORS								X	C1.DEV1.FINANCE.*.ENVRNMGR

**Note:** User access to data sets is indicated with an X. The user must have read access to the data sets in order to access actions.

See 4.4.1.1, "User Abbreviations" for a description of column headings.

### 4.4.4 Defining Security Rules for Your Package Actions Panel

The following three steps explain how to set up security for the Package Actions panel at your site.

1. Plan a table that lists the Package Actions panel menu items in the far left column and the users at your site along the top row. In the following example, an X indicates that a user has access to a menu item.

Menu Item	Appl Pgmr	Appl Mgr	Other Depts	QA	Prod Ctl	Tech Supp	Audit	Admin
DISPLAY	X	X	X	X	X	X		X
CREATE								X
MODIFY	X	X				X		X
CAST								X
REVIEW								X
EXECUTE	X			X	X	X		X
BACKOUT	X			X	X	X		X
COMMIT	X			X	X	X		X
UTILITY	X			X	X	X		X

2. Determine the format for your Package Actions panel security rules. In order to generate a pseudo data set name in the format 'C1'.PACKAGE'.menuitem.pkgsubfc.pkgid, you must set up the NAMEQU entry in the Name Equates Table using this format:

```
C          1      1          7 7
1          0      6          0 2
```

```
NAMEQU PACKAGE_ACTIONS,
        L1=('C1'),
        L2=('PACKAGE'),
        L3=(menuitem),
        L4=(pkgsubfc),
        L5=(pkgid)
```

3. In the tables below, determine the pseudo data set names based on the format described in Step 2 and the menu items indicated with an X in Step 1.

Menu Item	AP	AM	OD	QA	PC	TS	AU	AD	Data Set Names
DISPLAY	X	X	X	X	X	X		X	C1.PACKAGE.DISPLAY.APPROVER.PKG001
CREATE								X	C1.PACKAGE.CREATE.BUILD.PKG001
MODIFY	X	X				X		X	C1.PACKAGE.MODIFY.IMPORT.PKG001

Menu Item	AP	AM	OD	QA	PC	TS	AU	AD	Data Set Names
CAST								X	C1.PACKAGE.CAST.CAST.PKG001
REVIEW								X	C1.PACKAGE.REVIEW.DENY.PKG001
EXECUTE					X	X		X	C1.PACKAGE.EXECUTE.EXECUTE.PKG001
BACKOUT	X	X		X	X	X		X	C1.PACKAGE.BACKOUT.BACKOUT.PKG001
COMMIT	X	X		X	X	X		X	C1.PACKAGE.COMMIT.COMMIT.PKG001
UTILITY				X	X	X		X	C1.PACKAGE.UTILITY.EXPORT.PKG001

**Note:** User access to data sets is indicated with an X. The user must have READ access to the data sets in order to access to the Package Actions panel actions.  
See 4.4.1.1, “User Abbreviations” for a description of column headings.

## 4.4.5 Defining Rules for Your Site Security Package

Before using ESI, you must write rules to correspond to Formats ENVIRONMENT\_ACCESS through PACKAGE\_ACTIONS. You can customize these formats to conform to your site security conventions, to alter the names generated at each control point, or to change the authority level and class.

Endevor ESI lets you map Endevor entities to your site security package. Different security packages such as RACF, *eTrust* CA-ACF2, and *eTrust* CA-Top Secret have different approaches to implementing security. You need to understand the approach used by your site security package before attempting to map Endevor entities to your site security package.

### 4.4.5.1 What Happens if Site Security Rules Are Not Defined

All site security packages (RACF, *eTrust* CA-ACF2, and *eTrust* CA-Top Secret) deny access if security rules are not defined.

### 4.4.5.2 Guidelines for Writing Security Rules

Computer Associates recommends that you consider the following guidelines when writing security rules:

- Your security rules should conform to your existing site naming conventions.
- You should define action authorities based on pseudo data set rules rather than SAF authorities.
- Simplify rule definitions by omitting unnecessary format levels.
- Avoid inadvertently creating physical data sets when working with pseudo data sets.
- You should do ONE of the following:

- Use separate naming conventions for physical data sets and pseudo data sets.
- Define a different security class for pseudo data sets (see 4.2.3.3, “Defining a Class Other Than Data Set with RACF” on page 4-21).
- If the authorization values in the SAF interface are customized you should check and possibly modify the Endeavor authorization mapping (the FUNCEQU entries).

## 4.4.6 What to Do After You Fill Out Your ESI Worksheet

The following sections describe the tasks you should perform after you fill out your ESI worksheet:

- Developing ESI profiles
- Identifying user IDs requiring access
- Customizing your NAMEQU entries

### 4.4.6.1 Developing ESI Profiles

Use the pseudo data set names you created for the ESI worksheets to develop generic profiles. Determine whether you can combine any profiles before you submit the profiles to your site's security administrator.

For example, if you have three environments (DEV1, DEV2 and PROD) with three systems (FINANCE, PAYROLL and HUMNRCS) and you determine from the Action Initiation worksheet that QA has the authority to move elements for all three systems within the DEV1 and DEV2 environments. Assume that you originally coded the profiles as shown below:

```
C1.DEV1.FINANCE.*.MOVE
C1.DEV1.PAYROLL.*.MOVE
C1.DEV1.HUMNRCS.*.MOVE
C1.DEV2.FINANCE.*.MOVE
C1.DEV2.PAYROLL.*.MOVE
C1.DEV2.HUMNRCS.*.MOVE
```

Notice that in every case only the second and third qualifiers are different. The second qualifier must be specified because it defines the environment. The PROD environment is not included for QA access. The third qualifier, however, includes all systems so that you can make it generic with a wildcard character as shown below:

These three profiles:	Becomes one of these two profiles:
C1.DEV1.FINANCE.*.MOVE	C1.DEV1.*.*.MOVE
C1.DEV1.PAYROLL.*.MOVE	or
C1.DEV1.HUMNRCS.*.MOVE	C1.DEV1.-.*.MOVE
C1.DEV2.FINANCE.*.MOVE	C1.DEV2.*.*.MOVE
C1.DEV2.PAYROLL.*.MOVE	or
C1.DEV2.HUMNRCS.*.MOVE	C1.DEV2.-.*.MOVE

### 4.4.6.2 Identifying User Ids That Require Access

You need to supply the TSO IDs that are required to access the ESI rules/profiles you created. Whenever possible you should identify logical groups of IDs as shown in the following example:

```
all QA1* Ids
```

### 4.4.6.3 Customizing Your NAMEQU Entries

Use the ENVIRONMENT\_ACCESS through PACKAGE\_ACTIONS (extension ACTION\_INITIATION is optil) layouts you developed for the ESI worksheets to create NAMEQU entries for the Name Equates Table. Edit member BC1JNEQU of your iprfx.iqual.JCLLIB and modify the entries to reflect your choices for each of the formats.

The following format:	Is derived from:
ENVIRONMENT_ACCESS	The Environment Worksheet (Part 3)
PRIMARY_OPTIONS	The Primary Options Worksheet (Part 2)
BACKGROUND_OPTIONS	The Background Options Worksheet (Part 2)
ACTION_INITIATION	The Action Initiation Worksheet (Part 2)
ACTION_INITIATION (extension is optional)	The Action Initiation Worksheet (Part 2)
PACKAGE_ACTIONS	The Package Actions Worksheet (Part 1)

## 4.4.7 Assemble and Link the Name Equates Table

After you determine the ESIDFLTS, FUNCEQU and NAMEQU entries for your Name Equates Table you must complete the following four tasks:

- Define the rules to your site security package.
- Modify the BC1JNEQU member in your iprfx.iqual.JCLLIB.
- Assemble and link the modified table.
- Refresh the LINKLIST if the library is in the LINKLIST.

### 4.4.7.1 Defining Rules for Your Site Security Package

You should ensure that security rules are defined to your site security package. You can define security rules prior to enabling ESI. If, however, you enable ESI without defining security rules, all access is denied.

### 4.4.7.2 Modifying the BC1JNEQU Member

Edit member BC1JNEQU in your iprfx.igual.JCLLIB and modify the Name Equates Table entries to reflect the parameters you selected for your site. Ensure that the table name ESIDFLTS entry label matches the load module name created in the linkage step with the SYSLMOD DD statement.

### 4.4.7.3 Assembling and Linking the Modified Table

Submit member BC1JNEQU to assemble and link the new Name Equates Table into the authorized LINKLIST library.

### 4.4.7.4 Refreshing the LINKLIST

Perform an LLA REFRESH if your authorized library is in the LINKLIST. The following sample JCL assembles and links the Name Equates Table. The table name has been modified to NEWTNEQU. Note that the name on the ESIDFLTS entry label and the member name on the SYSLMOD DD statement match.

```
//* ( COPY JOBCARD )
/*****
//*
//* BC1JNEQU - THIS JOB IS USED TO CUSTOMIZE AND BUILD THE
//*          EXTERNAL SECURITY SYSTEM INTERFACE
//*          TABLE (BC1TNEQU).
//*
//* STEP1 WILL ASSEMBLE THE TABLE. THE TABLE NAME DEFAULTS TO
//* BC1TNEQU, OR CAN BE MODIFIED BY CODING A VALID MODULE NAME
//* ON THE ESIDFLTS ENTRY.
//*
//* STEP2 WILL LINKEDIT THE MEMBER AND PLACE IT IN YOUR
//*          INSTALLATION ENDEVOR-C1 CONLIB.
//*
//* THE FOLLOWING UPDATES MUST BE MADE TO THIS JCL BEFORE
//* IT CAN BE EXECUTED:
//*     1. TYPE OR COPY A VALID JOB CARD.
//*     2. VERIFY THAT THE OUTPUT DATA SET SPECIFIED ON THE SYSLMOD*
//*        DD STATEMENT IN STEP2 IS CORRECT. ALSO VERIFY THAT THE *
//*        OUTPUT MEMBER NAME IS THE SAME AS THE TABLE NAME THAT *
//*        IS GENERATED.
/*****
//STEP1 EXEC PGM=ASMA90,
//          REGION=2048K,
//          PARM='NODECK,OBJECT,NOTERM'
//SYSLIB DD DISP=SHR,DSN=iprfx.igual.SOURCE
//          DD DISP=SHR,DSN=SYS1.MACLIB
//SYSPRINT DD SYSOUT=*
//SYSPUNCH DD DUMMY
//SYSUT1 DD UNIT=tdisk,
//          SPACE=(TRK,(5,15))
//SYSLIN DD DSN=&&SYSLIN,
```

```

//          DISP=(NEW,PASS,DELETE),
//          UNIT=tdisk,
//          SPACE=(TRK,(3,5),RLSE),
//          DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSIN    DD  *
//*
          TITLE 'BC1TNEQU - EXTERNAL SECURITY INTERFACE TABLE.'
*****
*          DEFINE ESI DEFAULTS                                     *
*****
BC1TNEQU ESIDFLTS WARN=NO,                                       +
          TITLE='BC1TNEQU SECURITY INTERFACE TABLE',           +
          HEADER=YES,                                           +
          LATSIZ=2,                                             +
          DESC=6,                                               +
          ROUTCDE=11
*****
*          MAP E/OS390 AUTHORITIES TO SAF AUTHORITIES FOR       *
*          ACTION_INITIATION AND PACKAGE_ACTIONS FORMAT CALLS.  *
*          NOTE: ENVIRONMENT_ACCESS, PRIMARY_OPTIONS AND       *
*          FOREGROUND_OPTIONS FORMAT CALLS ALWAYS USE READ     *
*          AUTHORITY AND CANNOT BE MODIFIED.                   *
*****
          FUNCEQU SAFAUTH=READ,                                   +
          C1ACTNS=(ADD,ARCHIVE,DELETE,                           +
          DISPLAY,ENVRMGR,GENERATE,MOVE,                         +
          PBACKOUT,PCAST,PCOMMIT,PCREATE,PDISPLAY,              +
          PEXECUTE,PLIST,PMODIFY,PREVIEW,                        +
          PUTILITY,RETRIEVE,                                     +
          SIGNIN,SIGNOVR,UPDATE)
*****
*          SAMPLE SYNTAX OF OTHER SUPPORTED FUNCEQU AUTHORITIES *
*          LEVELS                                               *
*****
*          FUNCEQU SAFAUTH=NONE,                                  +
*          FUNCEQU SAFAUTH=UPDATE,                                +
*          FUNCEQU SAFAUTH=CONTROL,                              +
*          FUNCEQU SAFAUTH=ALTER,                                +
*****
*          END OF FUNCEQU SECTION                                *
*****
          FUNCEQU TYPE=END
          SPACE 2
*****
*          SPECIFY SAF DATASET NAME FORMATS                       *
*****
          NAMEQU ENVIRONMENT_ACCESS,                             +
          L1=('C1'),                                             +
          L2=('ENVIRON'),                                         +
          L3=(ENVIRONMENT)
          NAMEQU PRIMARY_OPTIONS,                                +
          L1=('C1'),                                             +
          L2=(ENVIRONMENT),                                       +
          L3=('PMENU'),                                           +

```

```

        L4=(MENUITEM)
NAMEQU FOREGROUND_OPTIONS,      +
        L1=('C1'),              +
        L2=(ENVIRONMENT),      +
        L3=('FORACTN'),        +
        L4=(MENUITEM)
NAMEQU ACTION_INITIATION,      +
        L1=('C1'),              +
        L2=(ENVIRONMENT),      +
        L3=(SYSTEM),           +
        L4=(SUBSYSTEM)
NAMEQU ACTION_INITIATION,      +
        L1=('C1'),              +
        L2=(MENUAUTH)
NAMEQU PACKAGE_ACTIONS,        +
        L1=('C1'),              +
        L2=('PACKAGE'),         +
        L3=(MENUITEM),         +
        L4=(PKGSUBFC),         +
        L5=(PKGID)
*****
*      SAMPLE SYNTAX OF OTHER SUPPORTED NAMEQU PACKAGE_ACTIONS      *
*      SYMBOLIC PARAMETERS                                           *
*****
*          LN=(PKGAPPGR),      +
*          LN=(PKGBOE),       +
*          LN=(PKGSHR),       +
*          LN=(PKGSTAT),      +
*          LN=(PKGTYPE),      +
*          WARN=NO
*****
*      END OF NAMEQU SECTION                                          *
*****
NAMEQU TYPE=END
END
//STEP2 EXEC PGM=IEWL,
//          PARM='LIST,NCAL,RENT,XREF,SIZE=(256K,64K)',
//          COND=(0,NE)
//SYSLMOD DD DISP=SHR,DSN=uprfx.uqua1.CONLIB(BC1TNEQU)
//SYSLIN DD DISP=(OLD,DELETE),DSN=&&SYSLIN
//SYSUT1 DD UNIT=t disk,
//          SPACE=(TRK,(5,15))
//SYSPRINT DD SYSOUT=*

```

At run time, Endeavor ESI dynamically loads the newly created Name Equates Table from an authorized library (either in LINKLIST or STEPLIB) to determine the RACROUTE request values to use.

## 4.4.8 Activating ESI Using the Endeavor C1DEFLT5 Table

The Endeavor C1DEFLT5 Table establishes the site, environment, and stage definitions for your installation. This table is an assembler macro that defines parameters specific to the Endeavor site as a whole, and to each environment and stage at the site.

### 4.4.8.1 The ACCSTBL Field

The ACCSTBL field contains the name of the Name Equates Table. The default value for the ACCSTBL field is BC1TNEQU.

### 4.4.8.2 The ESSI Field

The ESSI field has the following two purposes:

- To validate your purchase of ESI.
- To indicate if you want to use ESI security or native mode security.

The default value for the ESSI parameter is 0. This indicates that you did not purchase ESI and that you are using native mode security.

To indicate that you want to use ESI security you must enter a 'Y' or 'N' in the ESSI field in the C1DEFLT5 Table.

### 4.4.8.3 The PKGSEC Field

The PKGSEC keyword has three options:

- APPROVER — Indicates that Endeavor standard approver security should be used (pre-Release 3.8).
- ESI — Indicates that ESI interface is, with the exception of review processing which still requires the approver table (internal or external).
- MIGRATE — Allows both methods to be performed, with the Approver method having priority over the ESI method. Use this method if you're migrating to the ESI method.

**Package Security, External Approvers, and Approver Groups:** Endeavor allows you to have external approver groups and package security through ESI.

- APPROVER - Restricts package action through approver groups whether it is internal or external to Endeavor.
- ESI - Controls package actions with external security packages *eTrust CA-ACF2*, *eTrust CA-Top Secret* and RACF via the ESI interface.
- MIGRATE - Package actions are controlled by approver groups and/or ESI security.

How do these options differ?



---

APRVFLG=N,	APPROVAL PROCESSING (Y/N) *
<b>ESSI=Y,</b>	<b>ESI ENABLED</b> *
PKGSEC=Y,	PACKAGE CAST SECURITY *
<b>PKGSEC=ESI,</b>	<b>USE EXTERNAL SECURITY</b> *
RACFGRP=,	E/OS390 RACF GROUP NAME *
RACFPWD=,	E/OS390 RACF PASSWORD *
RACFUID=,	E/OS390 RACF USERID *

The first portion of the C1DEFLT5 Table (TYPE=MAIN) should be set up only once. For additional information about these fields, and about the C1DEFLT5 table in general, see the *Installation Guide* and the *User Guide*.

## 4.5 Testing ESI Security and Monitoring Warnings

Once you have successfully installed and enabled Endeavor ESI, you need to verify that ESI has been activated to monitor the security violations issued by Endeavor. See 4.6, “Using Endeavor ESI Warning Mode” on page 4-52 for more information about testing ESI.

### 4.5.1 How to Verify ESI Activation

To ensure Endeavor ESI is successfully enabled, display your site information.

```

----- Site Information from C1DEFULTS -----
Command ==>

Customer Name..... Computer Associates International, Inc.
----- Function Controls -----
Site ID..... 0      Access Table..... BCITNEQU      - Options -
Release..... B4000C  SMF Record Number. 000      ACM..... Y
Environments..... 2      Library System.... PV      DB2..... N
Userid Start..... 1      Library Program...      QuickEdit Y
Batch ID..... 0      VIO Unit..... SYSDA      ELINK.... N
SPFEDIT QNAME..... SPFEDIT  Work Unit..... SYSDA      ESI..... Y
SYSIEWL QNAME..... SYSIEWLP  Work Volser.....      INFO..... N
Authorized Tables. REQUIRED  Lines per Page.... 60      LIBENV... Y
Gen in place/SO... N      MODHLI.....      NETMAN... N
CA-LSERV JRNL SBS.      Signout on fetch.. Y      PDM..... Y
PITR Journal Grp..      ELINK XLTE TBL....      PROC..... Y
SYMBOLICS Table... ESYMBOLS  Mixed Format..... CCID COMMENT DESCRIPTION

(Press Enter for Next Panel)

```

```

----- Site Information from C1DEFULTS -----
Command ==>

----- Package Processing Options -----
Approval Required.... Y  Cast Security..... Y  Security.. ESI
Foreground Execution.. Y  Component Validation.. 0
High-level Index for Generated Remote Pkg Ship JCL...

----- Control Data Set Names -----
Element Catalog..... CA.ENDEAVOR.ELMCATL
Package Control File..... CA.ENDEAVOR.PACKAGE
Installation Macro Library. CA.ENDEAVOR.MACLIB
CCID Validation Data Set...
ACM Index Root Data Set.... CA.ENDEAVOR.ACMROOT
ACM Index Xref Data Set.... CA.ENDEAVOR.ACMXREF

----- CA-7 Interface Values -----
CA-7 Region CCI Nodename... TSONODE
JCL Data Set Index Number..
JCL Data Set Index Symbol.. &ENDEAVOR
JCL Data Set Name..... CA.ENDEAVOR.JCLLIB

(Press Enter for Previous Panel)

```

Endeavor ESI is enabled if Y appears in the field. The Access Table field contains the name of the Name Equates Table and the PKGSEC field is set to Y.

For more information about the site information screen, see the *Administration Guide*.

## 4.5.2 Monitoring Security Violations

You can monitor security violations using ESI warning mode or the CONRPT40 and CONRPT41 reports. Warning mode violations are recorded in the ESI warning report. See 4.6, “Using Endeavor ESI Warning Mode” on page 4-52 for more information about warning mode.

The CONRPT40 and CONRPT41 reports record attempts by users to perform unauthorized actions. See the *User Guide* for more information about using CONRPT40 and CONRPT41, as well as the *Reports Guide*.

## 4.6 Using Endeavor ESI Warning Mode

Endeavor ESI Warning Mode allows you to test your security implementation without denying users access to Endeavor objects. If access rules are not coded Endeavor ordinarily denies access. When using Warning Mode, access to resources is allowed even if your site security package (RACF, *eTrust* CA-ACF2, or *eTrust* CA-Top Secret) indicates that access should be denied. You should use ESI Warning Mode before writing security rules or as an initial test for your security rules.

When your security system identifies a security exception and ESI Warning Mode is enabled, a System Management Facility (SMF) record records the event. You can then use the ESI Exception Warning report to format and print the ESI warning SMF records in a convenient and easy-to-read report.

SMF records are always written to the SMF data sets (SYS1.MANx). In addition, you can send records to a sequential data set by allocating a data set to the DDname EN\$SMESI. This automatically writes the SMF records to the SMF data set as well the sequential data set.

The ESI Exception Warning report summarizes the SMF records written to the SMF and the sequential data sets. The report data is summarized by entity name, entity format type, entity class, user ID, event date and time. The original return codes and reason codes are listed as well as a summary report including each entity name and all the exceptions associated with the entity. The Exception Warning report also includes information about each Name Equates Table encountered.

The following JCL example shows how to execute an Exception Warning report:

```
//ESIWREPT EXEC PRM=NDVRC1,PARM='ENRASW00',REGION=4096K
//STEPLIB DD DISP=SHR,DSN=iprfx.igual.loadlib
//EN$SMESI DD DISP=SHR,DSN=SMF.data.set
//EN$SMRPT DD SYSOUT=*,DCB=(LRECL=133,BLKSIZE=6118,RECFM=FBA)
```

In this example, the program NDVRC1 invokes the exception report ENRASW00. EN\$SMESI identifies the input SMF records and EN\$SMRPT identifies the output data set.

### 4.6.1 ESI Defaults (ESIDFLTS) Macro

The ESI Defaults macro (ESIDFLTS) allows you to specify how you manage ESI diagnostics. You can use the ESIDFLTS macro to write a diagnostic trace, to improve performance and to enable warning mode.

## 4.6.2 Enabling ESI Warning Mode

To enable ESI warning mode you must specify `WARN=YES` in the Name Equates Table. The keyword is specified on either the `ESIDFLTS` or `NAMEQU` entries. The `ESIDFLTS` value affects the entire table. The `NAMEQU` value only affects the entry for which it is specified and overrides the `ESIDFLTS` value.

## 4.7 The Endeavor ESI Trace Facility

The Endeavor ESI Trace Facility helps you to determine if security is functioning as desired at your site. When the Trace Facility is activated every security call to ESI writes a trace record. The Trace DDname determines where the data is sent. You can use the Trace Facility to perform the following tasks:

- Ensure that the format of the entity name is correct.
- Review the results of a SAF request (a return code or a reason code).

### 4.7.1 Ensuring the Correct Format of the Pseudo Data Set Name

You can examine a trace record to determine if the format of the entity name is correct. The following sample trace record shows a highlighted pseudo data set name (Entity=C1.ENVIRON.PROD).

```

ENCS001I: Using BC1TNEQU table entitled 'ENDEAVOR NAMEQU Table 4.0'
ENCS001I: The table was assembled on 07/18/00 at 12.09
ENCS001I: ESI defaults:
ENCS001I: ESIDFLTS DESC=(6),
ENCS001I:     ROUTCDE=(11),
ENCS001I:     WARN=YES,
ENCS001I:     HEADER=YES,
ENCS001I:     LATSIZE=10,
ENCS001I:     TITLE='ENDEAVOR NAMEQU Table 4.0'
ENCS001I: Function authorization equates:
ENCS001I: FUNCEQU SAFAUTH=NONE,
ENCS001I:     C1ACTNS=(PRINT,TRANSFER,RESTORE,SIGNIN) 1
ENCS001I: FUNCEQU SAFAUTH=CONTROL,
ENCS001I:     C1ACTNS=(ARCHIVE,DELETE,DISPLAY,MOVE,RETRIEVE,SIGNOVR),
ENCS001I: FUNCEQU SAFAUTH=UPDATE,
ENCS001I:     C1ACTNS=(ADD,GENERATE,UPDATE)
ENCS001I: FUNCEQU SAFAUTH=ALTER,
ENCS001I:     C1ACTNS=(ENVRNMGR)
ENCS001I: Format definitions:
ENCS001I: NAMEQU ENVIRONMENT_ACCESS,
ENCS001I:     CLASS='$ENDEAVOR',
ENCS001I:     WARN=NO,
ENCS001I:     LOG=NONE,
ENCS001I:     L1=('ENDEAVOR'),
ENCS001I:     L2=('CLASS=$ENDEAVOR')
ENCS001I:     L3=('ENVIRONMENT_ACCESS_TEST'),
ENCS001I:     L4=('ENVIRONMENT=',ENVIRONMENT)
ENCS101I Format=0001 Pass=0000 Auth=READ ACEE=00000000
ENCS101I Class=$ENDEAVOR Log=NONE
ENCS101I Scale=0....+....1....+....2....+....3....+....4....+....5....+....6
ENCS101I Entity=ENDEAVOR.CLASS=$ENDEAVOR.ENVIRONMENT_ACCESS_TEST.ENVIRONMENT=D
      ....+....7....+....8....+....9....+....0....+....1....+....2
      EV
ENCS101I User DA2DM47 access is denied from SAF
ENCS101I RACROUTE RC=0008 RACHECK RC=0008 REASON=0000

```

```

ENCS101I Format=0001 Pass=0000 Auth=READ ACEE=00000000
ENCS101I Class=$ENDEVOR Log=NONE
ENCS101I Scale=0....+....1....+....2....+....3....+....4....+....5....+....6
ENCS101I Entity=ENDEVOR.CLASS=$ENDEVOR.ENVIRONMENT_ACCESS_TEST.ENVIRONMENT=B
          ....+....7....+....8....+....9....+....0....+....1....+....2
          ST
ENCS101I User DA2DM47 access is allowed from SAF

```

**1 CAUTION:**

While the **TRANSFER** and **RESTORE** functions are not valid for the **SAF=NONE** statement, they must be included for upward compatibility purposes.

## 4.7.2 Reviewing RACROUTE Request Return Codes

You can examine a trace record to review RACROUTE request return codes. The previous sample trace record includes highlighted RACROUTE return codes (RACROUTE(0000) RACHECK(0000) REASON(0000)).

## 4.7.3 The Trace Record Format

The Trace Facility trace records have the following format:

```

FORMAT=n
PASS=n
AUTH=n
ACEE=n
CLASS=DATASET
ENTITY=(Ln...)
USER=userid
ACCESS=(ALLOWED/DISALLOWED)
FROM=(SAF/LAT)
in WARN mode
RACROUTE RC=n
RACHECK RC=n
REASON RC=n

```

**Where:**

FORMAT= <i>n</i>	Specifies the NAMEQU format, where: <ul style="list-style-type: none"> <li>■ FORMAT1 is ENVIRONMENT_ACCESS</li> <li>■ FORMAT2 is PRIMARY_OPTIONS</li> <li>■ FORMAT3 is FOREGROUND_OPTIONS</li> <li>■ FORMAT4 and FORMAT5 are ACTION_INITIATION</li> <li>■ FORMAT6 is PACKAGE_ACTIONS</li> </ul>
PASS= <i>n</i>	Always 0 or 1.
AUTH= <i>auth</i> <b>1</b>	Specifies the requested authorization value as defined by the FUNCEQU macro: 'NONE' 'READ' 'UPDT' (UPDATE) 'CNTL' (CONTROL) 'ALTR' (ALTER)

---

<b>Where:</b>	
ACEE= <i>n</i>	For TSO environments this value is usually 0. For ROSCOE environments this value is the virtual storage address of the ACEE passed to the SAF interface.
CLASS= <i>class name</i>	Specifies the resource name security class. The default is class name.
ENTITY= <i>entity name</i>	Specifies the pseudo data set name passed to SAF.
USER= <i>userid</i>	Specifies a user ID for verification.
ACCESS=( <i>ALLOWED/DISALLOWED</i> )	Specifies whether access to the entity is allowed or disallowed.
FROM=( <i>SAF/LAT</i> )	Specifies either <i>SAF</i> or <i>LAT</i> .
in WARN mode	Displayed only when running in warning mode. Specifies that a user is allowed access because warning mode is turned on.
RACROUTE RC= <i>n</i>	Specifies the return code from the RACROUTE request: 0 = Permit request Non-zero = Fail request
RACHECK RC= <i>n</i>	Specifies the return code from the RACHECK request (internal to RACROUTE).
REASON= <i>n</i>	Specifies the reason code from the RACHECK.
<b>Note:</b>	
<b>1</b> : For <i>eTrust CA-ACF2</i> , and <i>eTrust CA-Top Secret</i> users, the AUTH value is alternately mapped to the appropriate value.	

---

## 4.7.4 Using the Trace Facility

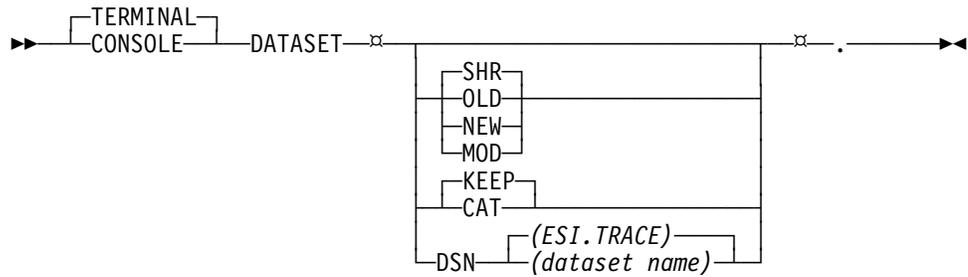
You must enable Endeavor ESI and have access to the Endeavor CLIST Library in order to use the ESITRACE command or DD statement. If you do not have access to the CLIST Library you can use the TSO ALLOCATE command. See TSO ALLOCATE Command Tasks for information about using the TSO ALLOCATE command.

### 4.7.4.1 Accessing the CLIST Library

See the *Installation Guide* to find out how to access the Endeavor CLIST Library.

### 4.7.4.2 ESITRACE Command in Foreground Mode

The ESITRACE command makes the Trace Facility easy-to-use. The ESITRACE syntax is shown below.

**ESITRACE Syntax:**

Use the following ESITRACE parameters to activate and deactivate the ESI Trace Facility in foreground mode.

**Start**

Allocates the trace data set which activates the Trace Facility. **START** is the default.

**Stop**

Deallocates the trace data set which deactivates the Trace Facility.

**TERMINAL**

Directs the trace output to your terminal. **TERMINAL** is the default.

**CONSOLE**

Directs the trace output to the operator's console.

**DATASET**

Allocates the trace output to the data set identified by the DSN parameter:

**SHR**

Allocates the trace data set with share status. **SHR** is the default.

**OLD**

Allocates the trace data set exclusively.

**NEW**

Allocates the trace data set with the name specified in the DSN parameter.

**MOD**

Writes new trace records at the end of the trace data set (EN\$TRESI).

**KEEP**

Keeps the trace data set when deallocated. **KEEP** is the default.

**CAT**

Catalogs the trace data set (EN\$TRESI) when deallocated.

**DSN (data set name)**

Identifies the name of the trace data set and is ignored for **CONSOLE** or **TERMINAL** allocations. The DSN follows the rules for TSO data set names.

**ESI.TRACE**

The default name of the trace data set.

*dataset name*

The user specified trace data set name.

**4.7.4.3 Running ESI Trace**

You can write trace records to a terminal, the operator's console or to data sets as shown in the following three examples:

1. Use the following CLIST parameter to start the Trace Facility and send all trace records to your terminal by default:

```
%ESITRACE START
```

2. Use the following CLIST parameter to start the Trace Facility and write trace records to the operator's console:

```
%ESITRACE CONSOLE START
```

3. Use the following CLIST parameter to start the Trace Facility and write all trace records to the data set named userid.ESI.TRACE:

```
%ESITRACE DATASET DSN (ESI.TRACE) NEW CATALOG
```

**Note:** You must preface the ESITRACE command with 'TSO' if you enter commands from an ISPF screen.

**4.7.4.4 Deactivating the Trace Facility**

You should deactivate the Trace Facility after you have determined that Endeavor ESI is properly configured. Use the following CLIST parameter to deactivate the Trace Facility.

```
%ESITRACE STOP
```

**4.7.4.5 TSO ALLOCATE Command Tasks**

The following table includes examples that show how you can use the TSO ALLOCATE and FREE commands to enable and disable the ESITRACE facility:

<b>To perform the following task:</b>	<b>Enter the following TSO ALLOCATE/FREE command:</b>
Start the Trace Facility	ALLOC DD (EN\$TRESI) DA(MY.ESI.DATASET) NEW CAT SPACE(1 1) CYLINDERS LRECL(133) BLKSIZE(6118) RECFM(FBA) UNIT(SYSDA) VOL(TSO001)
Allocate a data set to your terminal	ALLOC DD (EN\$TRESI) DA(*) SHR

<b>To perform the following task:</b>	<b>Enter the following TSO ALLOCATE/FREE command:</b>
Allocate an existing data set and append records	ALLOC DD (EN\$TRESI) DA(MY.ESI.TRACE.DATASET) MOD
Write trace data to the operator's console	ALLOC DD (EN\$TRESI) DUMMY SHR
Stop the Trace Facility	FREE DD (EN\$TRESI)

For more information about the TSO ALLOCATE command see your TSO documentation.

**Note:** If you allocate the ESI trace using the TSO ALLOCATE command or ISPF panels, you must use the following the DCB parameters:

- RECFM (FBA)
- LRECL (133)
- BLKSIZE (multiples of 133)

#### 4.7.4.6 ESI Trace in Batch Mode

You can activate the ESI Trace facility for batch processing by including the following DD statement in your execution JCL.

```
//EN$TRESI DD SYSOUT=*
```

You can write trace information to a data set by putting the following DD statement in your execution JCL.

```
//EN$TRESI DD DSN=data.set.name,DISP=SHR
```

Where DCB attributes match the ones noted above.



## **Appendix A. Security Worksheets**

---

## A.1 Environment Security Control Worksheet

Use the worksheets in this appendix to define security rules for your site.

### A.1.1 Part 1

In the table below, label and map your environments.


### A.1.2 Part 2

Enter environment names and indicate access with an "X" in the appropriate boxes.

Env \ User								
ENV1:								
ENV2:								
ENV3:								
ENV4:								

### A.1.3 Part 3

In the spaces below, determine the format for your Environment Rule.

(L1)\_\_\_\_\_ (L2)\_\_\_\_\_ (L3)\_\_\_\_\_

### A.1.4 Part 4

In the spaces below, determine pseudo dataset names based on the environment rule in Part 3 and the environments entered in Part 2. For example:

C1.ENVIRON.PROD

User must have READ access to this dataset in order to have environment access.

## A.2 Primary Options Security Control Worksheet

### A.2.1 Part 1

Environment: \_\_\_\_\_

System: \_\_\_\_\_

Subsystem: \_\_\_\_\_

### A.2.2 Part 2

In the table below, indicate the user's access to the Primary Options Menu item by marking an "X" in the appropriate box.

Primary Menu Item\User										
DISPLAY										
FOREGRND										
BATCH										
PACKAGE										
USER										
ENVRMENT										
UNLOAD										
RELOAD										
BATCHPKG										

### A.2.3 Part 3

In the spaces below, determine the format for your Primary Options Rule.

(L1)\_\_\_\_\_ (L2)\_\_\_\_\_ (L3)\_\_\_\_\_ (L4)\_\_\_\_\_

### A.2.4 Part 4

In the spaces below, determine pseudo dataset names; for example:

C1.PROD.PMENU.DISPLAY

User must have READ access to the dataset in order for this option to appear on Primary Options Menu.

## A.3 Foreground Options Security Control Worksheet

### A.3.1 Part 1

Environment: \_\_\_\_\_

System: \_\_\_\_\_

Subsystem: \_\_\_\_\_

### A.3.2 Part 2

In the table below, indicate the user's access to the Foreground Options Menu item by marking an "X" in the appropriate box.

<b>Frgd Menu Item\User</b>										
DISPLAY										
ADDUPDT										
RETRIEVE										
GENERATE										
MOVE										
DELETE										
SIGNIN										
PRINT										
BATCHPKG										

### A.3.3 Part 3

In the spaces below, determine the format for your Foreground Options Rule.

(L1)\_\_\_\_\_ (L2)\_\_\_\_\_ (L3)\_\_\_\_\_ (L4)\_\_\_\_\_

### A.3.4 Part 4

In the spaces below, determine pseudo dataset names; for example:

C1.PROD.FOREACTN.GENERATE

User must have READ access to the dataset in order for this option to appear on Primary Options Menu.

## A.4 Action Initiation Security Control Worksheet

### A.4.1 Part 1

Environment: \_\_\_\_\_

System: \_\_\_\_\_

Subsystem: \_\_\_\_\_

### A.4.2 Part 2

In the table below, indicate the user's authority to perform an action for the stage indicated by marking an "X" in the appropriate box.

Action Menu Item\User										
DISPLAY										
ADD										
RETRIEVE										
UPDATE										
GENERATE										
MOVE										
DELETE										
SIGNOVR										
ARCHIVE										
ENVRNMGR										

### A.4.3 Part 3

In the spaces below, determine the format of your Action Initiation Rule.

(L1)\_\_\_\_\_ (L2)\_\_\_\_\_ (L3)\_\_\_\_\_ (L4)\_\_\_\_\_ (L5)\_\_\_\_\_

### A.4.4 Part 4

In the spaces below, determine pseudo dataset names; for example:

C1.PROD.FINANCE\*.MOVE

User must have READ access to the dataset in order for this option to appear on Primary Options Menu.

## A.5 Package Actions Security Control Worksheet

### A.5.1 Part 1

In the table below, indicate the user's authority to perform an action for the stage indicated by marking an "X" in the appropriate box.

<b>Package Menu Item Subfunction\User</b>										
DISPLAY										
Blank										
Backout										
Approvers										
SCL										
Reports										
CREATE										
Build										
Import										
Edit										
Copy										
LIST*										
MODIFY										
Build										
Import										
Edit										
Copy										
CAST										
Cast										
SCL										
REVIEW										
Blank										
Deny										
Approve										

Package Menu Item Subfunction\User										
List										
Action Summary										
SHIP										
List										
Ship										
BACKOUT										
Backout										
Display										
Backin										
COMMIT										
Commit										
UTILITY										
Display										
Blank										
Backout										
Approvers										
SCL										
Reports										
Export										
Reset										
Delete										
<p><b>Note:</b></p> <ul style="list-style-type: none"> <li>■ <b>LIST</b> does not display as a menu item, but it is used by Endeavor to build available package lists.</li> <li>■ There is no security for the Notes subfunction in Release 3.8.</li> </ul>										

## A.5.2 Part 2

In the spaces below, determine the format of your Package Actions Rule.

(L1)\_\_\_\_\_ (L2)\_\_\_\_\_ (L3)\_\_\_\_\_ (L4)\_\_\_\_\_ (L5)\_\_\_\_\_

## A.5.3 Part 4

In the spaces below, determine pseudo dataset names; for example:

C1.PACKAGE.CREATE.\*.PKG\*

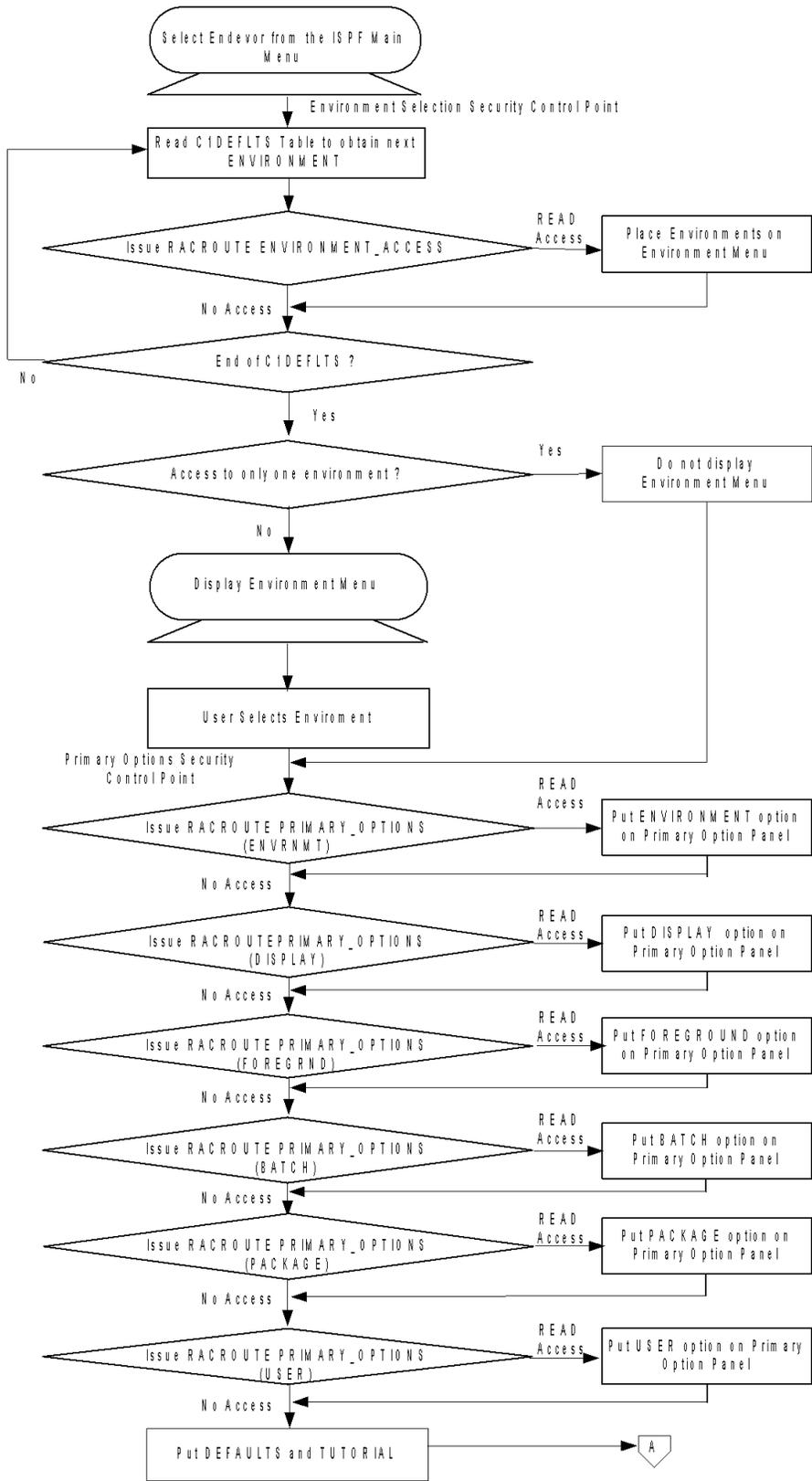
User must have READ access to the dataset in order for this option to appear on Primary Options Menu.

## **Appendix B. Endeavor ESI Logic Flow**

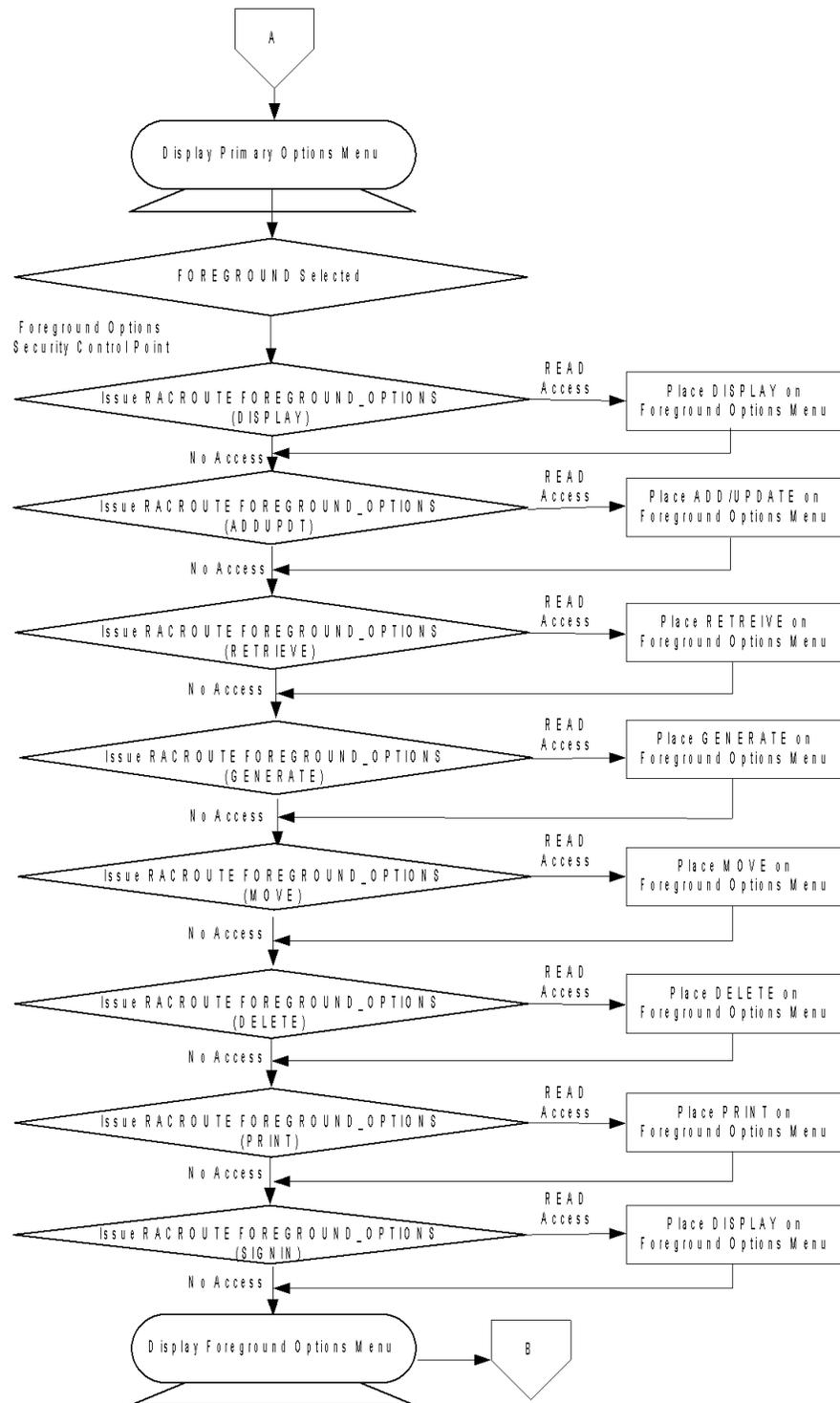
---

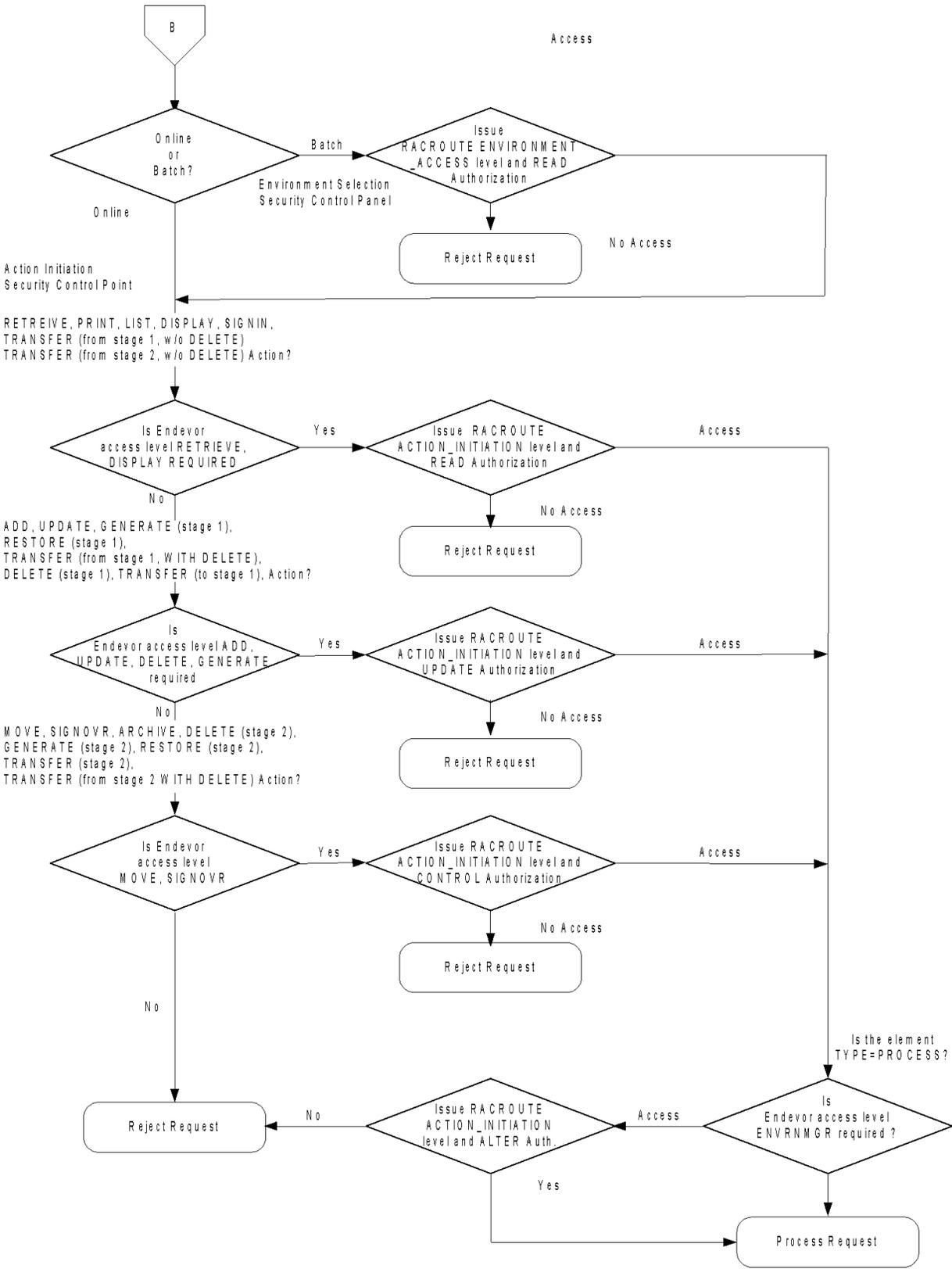
## **B.1 Endeavor ESI Logic Flow Diagram**

The following diagram shows the logic flow of Endeavor security decisions and FORMAT name generation. Please note that this flow chart illustrates default settings. You can alter these settings with MACRO definitions.



## B.1 Endeavor ESI Logic Flow Diagram







# Index

---

## A

- Access security table
  - assembling and link-editing 3-13
  - CONSDEF macros 3-11—3-13
  - defining 3-11
  - overview 3-11
  - processing flow explained 3-8
  - type=user macro parameters 3-12
  - updating defaults table 3-13
- Action initiations
  - data set 4-27
  - defining security rules 4-38—4-39
  - determining pseudo data set names 4-39
  - security control point
    - example 4-28
    - NAMEQU values 4-27
    - overview 4-27
    - pseudo data set format correlation 4-3
    - sample rules 4-27—4-28
  - security worksheet A-5
- Actions
  - access levels and codes for 3-18
    - See also* Sysdef=( ( macro
  - allowing and restricting 3-8
  - securing with FUNCEQU 4-12
- Alternate ID support
  - data sets controlled 2-2
  - data sets not controlled 2-3
  - using 2-2

## B

- BC1JACCT 3-13—3-14
- BC1JNEQU 4-44—4-46
- BC1JRSCT 3-25—3-26
- BC1JUSRT 3-20—3-21

## C

- C1ACTNS
  - using to omit RACROUTE requests 4-11
- C1DEFLT5
  - activating data set protection 2-4
  - activating ESI 4-47
  - changing name equates table name 4-48
  - external approvers 4-47
  - modifying 3-27
  - package security 4-47
  - updating for
    - access security table 3-13
    - resource security table 3-25
    - user security table 3-20
- CLIST
  - activating trace facility 4-56
  - deactivating trace facility 4-58
  - selecting trace record output destination 4-58
- CONSDEF macros
  - coding conventions 3-10
  - defining security tables
    - access 3-11—3-13
    - resource 3-22—3-24
    - user 3-15—3-20
  - order of security definitions 3-10
  - sysdef=( ( macro
    - See* Sysdef=( ( macro
  - type=user macro
    - See* Type=user macro
- Control points
  - access control diagram 1-4
  - changing names generated by 4-14—4-16
  - listed 3-5—3-6
  - overview 1-3
  - processing flow diagram 3-3

---

## D

### Data set security

- activating 2-4
- alternate ID support
  - data sets controlled 2-2
  - data sets not controlled 2-3
  - using 2-2
- approaches 1-2, 1-7
- CIDEFLTS 2-4
- files protected 2-2
- implementation steps listed 1-9
- program pathing 2-5
- relationship diagram 1-2
- TYPE=MAIN macro 2-4

### Data sets

- ACTION\_INITIATION 4-27
- defining name formats 4-13
- ENVIRONMENT\_ACCESS 4-25
- PACKAGE\_ACTIONS 4-30
- PRIMARY\_OPTIONS 4-26
- pseudo
  - See* Pseudo data set

### Defaults table

- activating data set protection 2-4
- activating ESI 4-47
- changing name equates table name 4-48
- modifying 3-27
- updating for
  - access security table 3-13
  - resource security table 3-25
  - user security table 3-20

## E

### Environment

- defining security rules 4-35—4-36
- determining pseudo data set names 4-36
- security worksheet A-2

### Environment access control point

- example 4-25
- NAMEQU values 4-25
- overview 4-25
- pseudo data set format correlation 4-3
- sample rules 4-25

### Environment selection security control point

- See* Environment access control point

### ESI

- control points
  - See* Control points
- defaults entries (ESIDFLTS)
  - See* ESIDFLTS

### ESI (continued)

- determining data set name formats 4-13
- developing profiles 4-42
- diagnostics 4-8
- enabling 4-34
- external approvers 4-47
- function equates entries (FUNCEQU)
  - See* FUNCEQU
- logic flow diagram B-2
- look aside table (LAT) 4-8
- name equates entries (NAMEQU)
  - See* NAMEQU
- name equates table
  - See* Name equates table
- overview 4-2, 1-5
- package security 4-47
- pseudo data set
  - See* Pseudo data set
- security
  - monitoring violations of 4-51
  - processing model 4-4
- site security packages interaction diagram 1-5
- trace facility
  - See* Trace facility
- verifying activation 4-50
- warning mode 4-52
- worksheets
  - See* Security worksheets

### ESIDFLTS

- look aside table (LAT) 4-8
- managing ESI diagnostics 4-8
- overview 4-3
- parameters 4-8—4-9
- warning mode 4-52

### ESITRACE command

- batch mode 4-59
- foreground mode 4-56—4-58

### eTrust CA-ACF2

- defining security rules 4-41
- ESI interaction diagram 1-5
- mapping authorization values to 4-11
- program pathing 2-5
- RACF parameters 2-4
- security processing model 4-4

### eTrust CA-Top Secret

- defining security rules 4-41
- ESI interaction diagram 1-5
- mapping authorization values to 4-11
- program pathing 2-5
- RACF parameters 2-4
- security processing model 4-4

---

Exception warning reports 4-52  
Exit modules 4-5  
External approvers 4-47  
External security interface  
    *See* ESI

## F

Foreground options  
    panel  
        defining security rules 4-38  
    security control point  
        example 4-27  
        pseudo data set format correlation 4-3  
        sample rules 4-27  
    security worksheet A-4  
FORMAT1-5  
    new names 4-3  
    trace record parameter 4-55  
FREE command  
    manipulating trace facility 4-58  
FUNCEQU  
    auth values 4-10  
    cIaccess levels 4-11  
    changing mapping of access levels 4-10  
    examples  
        four level authorization 4-12  
        single level authorization 4-9  
    overview 4-3  
    SAF authorization 4-12  
    securing actions 4-12  
Functional security  
    approaches 1-3, 1-7  
    enabling 1-9  
    relationship diagram 1-2

## I

ISPTASK 2-6

## K

Keywords  
    compatible formats 4-17—4-18  
    definitions 4-18—4-21  
    package symbolics note 4-21  
    specifying substrings of 4-16

## L

Logic flow diagram B-2

Look aside table (LAT) 4-8  
    *See also* ESIDFLTS

## M

Moves  
    access levels and codes 3-19

## N

Name equates table  
    assembling and linking 4-43—4-46  
    changing name 4-48  
    coded entries 4-7—4-8  
    control point rule formats for  
        action initiations 4-27  
        environment access 4-25  
        package actions 4-30  
        primary options 4-26  
    defining  
        ESI diagnostics 4-8  
        SAF authorization levels 4-9  
        SAF name formats 4-13  
        security rule formats 4-23  
    enabling ESI warning mode 4-53  
    sample table 4-6—4-7  
NAMEQU  
    control point values for  
        action initiations 4-27  
        environment access 4-25  
        foreground options 4-26  
        package actions 4-30  
    creating ENTITY=dsname value 4-13  
    customizing entries 4-43  
    generating pseudo data set names for  
        action initiations 4-39  
        environment 4-36  
        package actions panel 4-40  
        primary options panel 4-37  
    overview 4-3  
    PACKAGE\_ACTION package symbolics 4-31  
    SAF name formats  
        changing control point generated  
            names 4-14—4-16  
        creating 4-13  
        creating an additional class category 4-21—4-23  
        specifying keyword substrings 4-16  
Native security  
    control points  
        *See* Control points  
    implementation steps listed 3-9

---

## Native security (*continued*)

- overview 1-4
- relationship diagram 1-2
- security tables
  - defining access 3-11—3-14
  - defining resource 3-22—3-26
  - defining user 3-15—3-21
  - defining, overview 3-9
  - modifying defaults table 3-27
  - processing of 3-8
  - user exit modules 3-7

## P

### Package actions

- data set 4-30
- ESI calls 4-32
- package symbolics note 4-31
- panel
  - defining security rules 4-40—4-41
  - determining pseudo data set names 4-40
- security control point
  - NAMEQU values 4-30
  - overview 4-30
  - pseudo data set format correlation 4-3
  - sample rules 4-30
  - security worksheet A-6

### Package security 4-47

### Package symbolics

- PACKAGE\_ACTION 4-31

### Primary options

- data set 4-26
- panel
  - defining security rules 4-37
  - determining pseudo data set names 4-37
- security control point
  - NAMEQU values 4-26
  - overview 4-26
  - pseudo data set format correlation 4-3
  - sample rules 4-26
  - security worksheet A-3

### Processing

- access levels and codes for
  - actions 3-18—3-19
  - moves 3-19—3-20
  - transfers 3-19—3-20
- security control points
  - See* Control points

### Program pathing

- diagram 2-6
- using eTrust CA-ACF2 and eTrust CA-Top Secret 2-5

## Pseudo data set

- determining names for
  - action initiations 4-39
  - environment 4-36
  - package actions panel 4-40
  - primary options panel 4-37
- developing ESI profiles 4-42
- format and control point correlation table 4-3
- function 4-2
- security rule formats 4-23
- use with trace facility 4-54

## R

### RACF

- creating an additional class category 4-21—4-23
- data set security activation parameters 2-4
- defining security rules 4-41
- ESI interaction diagram 1-5
- mapping authorization values to 4-11
- security processing model 4-4

### RACROUTE

- creating an additional class category 4-21—4-23
- requests
  - creating ENTITY=dsname value 4-13
  - default authorization value 4-28—4-30
  - omitting 4-11
  - reviewing return codes 4-55
  - security processing model 4-4
  - security rule formats 4-23

### Resource security table

- assembling and link-editing 3-25
- CONSDEF macros 3-23—3-24
- defining 3-22
- overview 3-22
- processing flow explained 3-8
- sysdef=( ( macro 3-24
- type=user parameters 3-23

## S

### SAF authorization levels

- changing mapping of access levels 4-10
- defining 4-9
- FUNCEQU authorization 4-12
- mapping authorization values to site security packages 4-11

### SAF name formats

- changing control point generated names 4-14—4-16
- creating an additional class category 4-21—4-23
- creating with NAMEQU 4-13

---

SAF name formats (*continued*)

- keywords
  - compatible formats 4-17—4-18
  - definitions 4-18—4-21
  - package symbolics note 4-21
  - specifying substrings of 4-16

Security

- control points
  - See* Control points
- defining rules for
  - action initiations 4-38—4-39
  - environments 4-35—4-36
  - foreground options panel 4-38
  - package actions panel 4-40—4-41
  - primary options panel 4-37
  - site security package 4-41
  - sites 4-35
- guidelines for writing rules 4-41
- monitoring violations of 4-51
- processing model 4-4
- tables
  - access 3-11—3-14
  - resource 3-22—3-26
  - user 3-15—3-21
- worksheets
  - See* Security worksheets

Security worksheets

- action initiation A-5
- environment A-2
- foreground options A-4
- package actions A-6
- preparing 4-34
- primary options A-3
- tasks to perform upon completion 4-42—4-43

Site security packages

- defining rules 4-41
- ESI interaction diagram 1-5
- program pathing 2-5
- security processing model 4-4

Sites

- defining security rules for
  - action initiations 4-38
  - environments 4-35
  - package actions panel 4-40
  - primary options panel 4-37
  - site security package 4-41

SMF records 4-52

Sysdef=( ( macro

- access security table 3-12—3-13
- resource security table 3-24
- user security table 3-17—3-20

System Authorization Facility

*See* SAF

System Management Facility records 4-52

## T

Trace facility

- activating 4-56
- deactivating 4-58
- ESITRACE command
  - See* ESITRACE command
- format of pseudo data set name 4-54
- reviewing RACROUTE request return codes 4-55
- running 4-58
- trace records
  - See* Trace records
- uses 4-54
- using in batch 4-59

Trace records

- format 4-55—4-56
- sample 4-54—4-55
- selecting output destination 4-58

Transfers

- access levels and codes 3-19

TSO ALLOCATE command

- manipulating trace facility 4-58

TSO user ID

- data sets controlled 2-3

TYPE=ENVRNMNT

- RSCETBL parameter 3-27
- use with security tables 3-27
- USERTBL parameter 3-27

TYPE=MAIN macro

- ACCSTBL parameter 3-27
- activating data set protection 2-4
- use with security tables 3-27

Type=user macro

- parameters for
  - access security table 3-12
  - resource security table 3-23
  - user security table 3-16
- sysdef=( ( macro
  - See* Sysdef=( ( macro

## U

User exit modules 3-7, 4-5

User security table

- assembling and link-editing 3-20
- CONSDEF macros 3-15—3-20
- defining 3-15

---

User security table (*continued*)

- overview 3-15
- processing flow explained 3-8
- sysdef=( ( macro 3-17—3-20
- type=user macro parameters 3-16

## **W**

Warning mode

- enabling 4-53
- ESIDEFLTS macro 4-52
- exception warning reports 4-52
- function 4-52
- SMF records 4-52