

CA-Top Secret[®]

User Guide
Release 3.0
VSE



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.

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.

Second Edition, September 2000

©1985-2000 Computer Associates International, Inc.
One Computer Associates Plaza, Islandia, NY 11749
All rights reserved.

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

Contents

About This Guide	xv
Chapter 1. Introducing CA-Top Secret	1-1
1.1 Why Use CA-Top Secret?	1-2
1.1.1 Purpose	1-2
1.1.2 Components and Features	1-2
1.2 How Does CA-Top Secret Work?	1-5
1.2.1 System Entry Protection	1-6
1.2.1.1 Types of ACIDs and How They're Structured	1-6
1.2.1.2 The Role of the Security Administrator	1-10
1.2.1.3 Passwords	1-12
1.2.1.4 ACID and Password Validation	1-13
1.2.1.5 Modes	1-13
1.2.2 What is a Facility?	1-14
1.2.3 What is a Resource?	1-14
1.2.4 What is a Field?	1-15
1.2.5 Control Options and Command Functions	1-15
1.2.5.1 What are Control Options?	1-15
1.2.5.2 What are Command Functions?	1-16
1.2.5.3 Entry Methods	1-18
1.2.6 Distributed Security Processing	1-18
1.2.6.1 What is the Command Propagation Facility (CPF)?	1-18
1.2.6.2 Synchronizing Information Across Nodes	1-19
1.2.6.3 Controlling Access From Remote Nodes	1-19
1.3 Where is Information Stored?	1-20
1.3.1 CA-Top Secret Files	1-20
1.3.1.1 Security File	1-20
1.3.1.2 Parameter File	1-20
1.3.1.3 Facility Matrix Table	1-20
1.3.1.4 Audit/Tracking File	1-20
1.3.1.5 Backup File	1-21
1.3.1.6 Recovery File	1-21
1.3.1.7 \$\$LOG\$\$ File	1-21
1.3.1.8 Command Propagation Files	1-21
1.3.2 Special Security Records	1-22
Chapter 2. Implementing CA-Top Secret	2-1
2.1 Summary of Implementation Steps	2-2
2.2 Implementing Phased Security	2-3
2.2.1 DORMANT Mode	2-3
2.2.2 WARN Mode	2-3
2.2.3 IMPLEMENT Mode	2-4
2.2.4 FAIL Mode	2-4
2.2.5 Using Concurrent Security Modes	2-5
2.3 Planning Emergency Procedures	2-7
2.3.1 Production Security Violations	2-7

2.3.2	Emergency ACIDs	2-7
2.3.3	CA-Top Secret Software Problems	2-8
2.3.4	Disaster Recovery	2-10
2.4	Implementing Default Protection	2-11
2.4.1	Default Data Set Protection	2-11
2.4.2	Migrating to FAIL Mode	2-11
2.5	Implementing by Facility	2-13
2.5.1	BATCH Considerations	2-14
2.5.2	CICS, DL/I, and CA-IDMS Considerations	2-17
2.5.3	Other Facilities Considerations	2-20
2.5.4	APPC/VSE Considerations	2-20
2.6	Developing Tape Protection	2-21
2.7	Protecting Special Resources	2-23
2.7.1	VSE Libraries	2-23
2.7.2	CPUs	2-23
2.7.3	Terminals	2-23
2.7.4	Record Level Protection (RLP)	2-24
2.7.5	Screen Level Protection (SLP)	2-24
2.8	Logging and Reporting Options	2-26
2.8.1	Logging Activity and Violations	2-27
2.8.2	Reporting Activity and Violations	2-27
2.8.3	Online Tracking of Activity and Violations	2-28
2.8.4	User Message and Violation Suppression	2-28
2.9	Defining Audit Requirements	2-30
2.9.1	Defining CA-Top Secret Auditors	2-30
2.9.2	Using CA-Top Secret Auditing Capabilities	2-31
2.10	Developing Security Maintenance Procedures	2-33
2.10.1	Verifying Change Requests	2-33
2.10.2	CA-Top Secret Software Maintenance	2-34
2.10.3	System Software Maintenance	2-35
Chapter 3. Violations, Logging, Reporting, and Auditing		3-1
3.1	Violations and Logging	3-2
3.2	Logging Security Events	3-4
3.2.1	TSSAUDIT	3-4
3.2.2	TSSCPR	3-4
3.2.3	TSSRECVR	3-5
3.2.4	TSSRPTST	3-5
3.2.5	TSSTRACK	3-5
3.3	Running Security Reports	3-6
3.3.1	TSSCHART	3-6
3.3.2	TSSUTIL	3-6
3.3.3	TSSCFILE	3-7
3.3.4	TSSREPORT and TSSREPORT2	3-7
3.4	Auditing	3-8
3.4.1	Auditing Users and Resources	3-9
Chapter 4. Setting Up and Modifying Your Security Environment		4-1
4.1	Creating Security Administrators	4-2
4.1.1	Adding Control ACIDs	4-3

4.1.1.1	How to Create the MSCA	4-3
4.1.1.2	How to Create an SCA	4-4
4.1.1.3	How to Create an LSCA	4-4
4.1.1.4	How to Create a ZCA	4-5
4.1.1.5	How to Create a VCA	4-5
4.1.1.6	How to Create a DCA	4-7
4.1.1.7	How To Create an Auditor	4-7
4.1.2	Assigning Administrative Authority	4-7
4.1.2.1	ACID Authorities	4-9
4.1.2.2	DATA Authorities	4-10
4.1.2.3	RESOURCE Authorities	4-11
4.1.2.4	FACILITY Authorities	4-13
4.1.2.5	MISC1 Authorities	4-14
4.1.2.6	MISC2 Authorities	4-15
4.1.2.7	MISC3 Authorities	4-16
4.1.2.8	MISC8 Authorities	4-16
4.1.2.9	MISC9 Authorities	4-17
4.1.2.10	SCOPE Authority	4-18
4.1.3	Removing Administrative Authorities	4-18
4.2	Identifying Users by Defining ACIDs	4-19
4.2.1	Adding Department, Division, and Zone ACIDs	4-19
4.2.1.1	How to Create a Zone	4-20
4.2.1.2	How to Create a Division	4-20
4.2.1.3	How to Create a Department	4-21
4.2.2	Creating User ACIDs	4-22
4.2.3	Using an Existing ACID to Create Another	4-23
4.2.4	Promoting or Demoting ACIDs	4-24
4.2.5	Deleting ACIDs	4-25
4.2.6	Setting ACID Expiration Dates	4-26
4.2.6.1	Deactivating an Expiration Date to Reactivate a User	4-26
4.2.6.2	Changing an Expiration Date	4-27
4.2.6.3	Setting Automatic Inactivity Suspensions	4-27
4.2.6.4	Setting Limited Duration Suspensions	4-27
4.3	Creating Profiles	4-29
4.4	Creating Groups	4-33
4.5	ACID Creation Requirements Chart	4-35
4.6	Displaying Information	4-36
4.6.1	Displaying ACID and Record Information	4-37
4.6.2	Displaying Access Authorization Information	4-39
4.6.3	Displaying Resource Ownership	4-40
4.6.4	Displaying Individual ACID Security Information	4-41
4.6.5	Displaying the Global Security Environment	4-41
4.7	Using Control Options to Change Your Security Environment	4-43
4.7.1	Restricted Control Options	4-44
4.7.2	Selecting Control Options Through the Parameter File	4-45
Chapter 5.	Password and System Entry Security	5-1
5.1	Password Security	5-2
5.1.1	Defining Passwords	5-3
5.1.1.1	How Password Creation Defaults Are Set	5-3

5.1.1.2	How to Define a Password for a New User	5-4
5.1.2	Changing Passwords	5-5
5.1.2.1	Reverifying a Changed Password	5-7
5.1.2.2	Maintaining Password History	5-8
5.1.2.3	Displaying a User's Password	5-8
5.1.2.4	Keeping Users From Changing Their Passwords	5-9
5.1.3	Specifying Optional Password Features	5-9
5.1.3.1	Changing the Restricted Password List	5-9
5.1.3.2	Using Masks to Define a Password	5-11
5.1.3.3	Using Random Password Generation	5-12
5.1.3.4	Forcing Password Validation By Mode	5-14
5.1.3.5	Selecting a Password Violation Threshold	5-15
5.1.4	Summary of Password Options	5-16
5.2	System Entry Security	5-17
5.2.1	Monitoring Entry Through Facility Authorization	5-17
5.2.2	Monitoring Entry by Time of Use	5-18
5.2.2.1	By Day of Week	5-18
5.2.2.2	By Time of Day	5-18
5.2.2.3	For a Length of Time	5-19
5.2.2.4	To Correct Time Zone Discrepancies	5-20
5.2.2.5	For Calendars or Time Records	5-20
5.2.3	Monitoring Entry by Device	5-21
5.2.3.1	How to Restrict Signons to Specific Devices	5-21
5.2.3.2	How to Control Multiple Signons	5-22
5.2.3.3	Default ACID Signon	5-23
5.2.3.4	Automatic Terminal Signon	5-23
5.2.3.5	How to Lock and Unlock Terminals	5-24
5.2.3.6	Automatic Locking	5-25
5.2.3.7	Manual Locking	5-26
5.2.3.8	Unlocking a Locked Terminal	5-26
5.2.3.9	How to Use Special Authentication Devices	5-26
5.2.4	Monitoring Entry by Job Validation	5-27
5.2.5	Monitoring Entry Within the Network	5-28
5.2.5.1	Defining Users to Multiple Nodes	5-28
5.2.5.2	Using the Secured Signon Feature	5-28
Chapter 6.	Resource Security Validation	6-1
6.1	Owning Resources	6-3
6.1.1	Assigning Ownership	6-4
6.1.1.1	Selecting Resource Owners	6-4
6.1.1.2	Using Generic Prefixing	6-5
6.1.1.3	Using Masking	6-9
6.1.1.4	How to Assign Ownership	6-13
6.1.2	Locating the Owner of a Resource	6-15
6.1.3	Transferring Resource Ownership	6-15
6.1.4	Removing Ownership	6-16
6.1.5	List of Predefined Resources	6-16
6.1.5.1	For VSE and/or MVS	6-17
6.1.5.2	For VSE	6-19
6.1.5.3	For VM	6-20

6.2	Controlling Access	6-21
6.2.1	Providing Default Protection	6-22
6.2.2	Authorizing Access	6-24
6.2.2.1	Making Resources Globally Accessible	6-25
6.2.2.2	Adding Resources to the ALL Record	6-25
6.2.2.3	Overriding Global Authorizations	6-26
6.3	Selective Revoke Command	6-27
6.3.1	Using Multiple Access Authorizations	6-27
6.3.2	Restricting Access	6-28
6.3.2.1	By Facility	6-29
6.3.2.2	By Program Path	6-29
6.3.2.3	By Access Levels	6-30
6.3.2.4	With the ACTION Keyword	6-33
6.3.3	Denying Access	6-34
6.4	The Security Validation Algorithm	6-36
6.4.1	How Does the Security Validation Algorithm Work?	6-36
6.4.2	How to Set Up Your Security Validation Algorithm	6-37
6.4.2.1	Specifying an AUTH Control Option Value	6-37
6.4.2.2	Using the ATTR Keyword	6-40
6.4.3	Processing Volume Requests	6-40
6.4.4	Processing Data Set Requests	6-42
6.4.5	Processing General Resource Requests	6-45
6.4.6	How Best Fit is Determined	6-46
Chapter 7. Protecting Resources		7-1
7.1	Protecting Volumes	7-4
7.1.1	How to Extend Default Protection	7-5
7.1.2	How to Use Generic Prefixing	7-5
7.1.3	How to Authorize Access	7-5
7.1.3.1	How to Assign Access Levels	7-6
7.1.3.2	How to Assign a Program Path	7-7
7.1.3.3	How to Access All Volumes	7-7
7.1.4	How to Bypass Volume Level Security	7-8
7.1.5	How to Request Only Volume Level Security	7-8
7.1.6	Using Volume Management Packages	7-9
7.1.6.1	Required Backup and Restore Access Authorizations	7-9
7.2	Protecting Data Sets	7-10
7.2.1	How to Extend Default Protection	7-10
7.2.2	How to Use Generic Prefixing	7-11
7.2.3	How to Use Masking	7-12
7.2.4	How to Authorize Access	7-13
7.2.4.1	How to Assign Access Levels	7-13
7.2.4.2	How to Assign a Program Path	7-15
7.2.4.3	How to Access All Data Sets	7-16
7.2.5	How to Bypass Data Set Level Security	7-17
7.3	Protecting Tape Volumes	7-18
7.3.1	Using a Tape Management Package	7-18
7.3.2	Securing Bypass Label Processing in OS/390	7-19
7.4	Data Set Vs. Volume Level Security	7-20
7.5	Protecting Terminals	7-21

7.5.1.1	How to Use Generic Prefixing	7-22
7.5.1.2	How to Authorize Access	7-23
7.5.1.3	How to Access All Terminals	7-24
7.6	Protecting Programs	7-25
7.6.1	How to Assign Default Protection	7-26
7.6.2	How to Use Generic Prefixing	7-26
7.6.3	How to Authorize Access	7-27
7.6.3.1	How to Assign a Program Path	7-27
7.6.3.2	How to Access All Programs	7-28
7.7	Protecting Applications	7-29
7.7.1	How to Use Generic Prefixing	7-29
7.7.2	How to Authorize Access	7-30
7.8	Protecting Installation-Defined Resources	7-32
7.8.1	How to Use Generic Prefixing	7-33
7.8.2	How to Authorize Access	7-34
7.8.2.1	How to Assign Access Levels	7-34
7.8.2.2	How to Assign a Program Path	7-35
7.9	Protecting Online Transactions	7-36
7.9.1	Implementing OTRAN Security	7-36
7.9.1.1	How to Set Up OTRAN Security	7-37
7.9.2	How to Use Generic Prefixing	7-38
7.9.2.1	How to Authorize Access	7-38
7.9.2.2	How to Assign a Program Path	7-39
7.9.3	Implementing LCF Security	7-39
7.9.3.1	How to Set Up LCF Security	7-40
7.9.3.2	How to Provide Default Protection	7-42
7.9.3.3	How to Bypass LCF Security	7-42
7.9.3.4	How to Use Generic Prefixing	7-43
7.9.3.5	How to Use Reverification	7-43
7.9.3.6	How to Access All Transactions	7-43
7.9.4	Implementing Screen Level Protection (SLP)	7-43
7.9.4.1	Gather Information	7-44
7.9.4.2	Enter Definitions	7-44
7.9.4.3	Permit Access to the Defined Maps	7-45
7.9.4.4	Enable Protection	7-45
Chapter 8.	Maintaining Special Security Records	8-1
8.1	Resource Descriptor Table (RDT) Record	8-2
8.1.1	Modifying the RDT	8-2
8.1.2	Maintaining the RDT	8-3
8.1.2.1	Defining a Resource to the RDT	8-3
8.1.2.2	Changing Values in the RDT	8-7
8.1.2.3	Removing a Resource in the RDT	8-8
8.1.2.4	Listing the RDT	8-9
8.1.3	Resource Masking	8-9
8.2	Field Descriptor Table (FDT) Record	8-11
8.2.1	Modifying the FDT	8-12
8.2.2	Maintaining the FDT	8-12
8.2.2.1	Defining New Fields to the FDT	8-13
8.2.2.2	Changing User-Defined Fields in the FDT	8-14

8.2.2.3	Removing Fields From the FDT	8-16
8.2.2.4	Listing the FDT	8-17
8.3	Static Data Table (SDT) Record	8-19
8.3.1	Defining Record Elements to the SDT	8-20
8.3.1.1	How to Define CALENDAR Records	8-20
8.3.1.2	How to Define MAP Records	8-23
8.3.1.3	How to Define MASK Records	8-24
8.3.1.4	How to Define RLP Records	8-25
8.3.1.5	How to Define SELECT Records	8-26
8.3.1.6	How to Define TIME Records	8-27
8.3.2	Replacing Data Elements in the SDT	8-28
8.3.2.1	How to Replace CALENDAR Records	8-28
8.3.2.2	How to Replace MAP Records	8-28
8.3.2.3	How to Replace MASK Records	8-28
8.3.2.4	How to Replace RLP Records	8-28
8.3.2.5	How to Replace SELECT Records	8-29
8.3.2.6	How to Replace TIME Records	8-29
8.3.3	Removing Fields From Records in the SDT	8-29
8.3.3.1	How to Remove Dates From CALENDAR Records	8-29
8.3.3.2	How to Remove Fields From MAP Records	8-29
8.3.3.3	How to Remove Fields From MASK Records	8-30
8.3.3.4	How to Remove Fields From RLP Records	8-30
8.3.3.5	How to Remove Fields From SELECT Records	8-30
8.3.3.6	How to Remove Times From TIME Records	8-30
8.3.4	Deleting Record Elements From the SDT	8-30
8.3.4.1	How to Delete CALENDAR Records	8-31
8.3.4.2	How to Delete MAP Records	8-31
8.3.4.3	How to Delete MASK Records	8-31
8.3.4.4	How to Delete RLP Records	8-31
8.3.4.5	How to Delete SELECT Records	8-31
8.3.4.6	How to Delete TIME Records	8-32
8.3.5	Listing Record Elements in the STD	8-32
8.3.5.1	How to List CALENDAR Records	8-32
8.3.5.2	How to List MAP Records	8-32
8.3.5.3	How to List MASK Records	8-32
8.3.5.4	How to List RLP Records	8-33
8.3.5.5	How to List SELECT Records	8-33
8.3.5.6	How to List TIME Records	8-33
8.4	Node Descriptor Table (NDT) Record	8-34
8.4.1	Defining Data to the NDT	8-35
8.4.2	Changing Values in the NDT	8-36
8.4.3	Removing Data From the NDT	8-37
8.4.4	Listing the NDT	8-38
8.5	ALL Record	8-39
8.6	APPCLU Record	8-40
8.6.1	Defining LINKIDs to the APPCLU Record	8-40
8.6.2	Changing Data in the APPCLU Record	8-42
8.6.3	Removing a Link From the APPCLU Record	8-43
8.6.4	Listing the APPCLU Record	8-43
8.7	AUDIT Record	8-44

8.7.1	Adding a Resource to the AUDIT Record	8-44
8.7.2	Removing a Resource From the AUDIT Record	8-44
8.7.3	Listing the AUDIT Record	8-45
8.8	Data Lookaside Facility (DLF) Record	8-46
8.8.1	Defining Data to the DLF Record	8-46
8.8.2	Removing Data From the DLF Record	8-47
8.8.3	Listing the DLF Record	8-47
Chapter 9.	Command Propagation Facility	9-1
9.1	Overview	9-2
9.1.1	Communication Components	9-2
9.1.2	CPF Architecture	9-3
9.1.2.1	Implicit and Explicit Targeting	9-3
9.1.2.2	Synchronous and Asynchronous Processing	9-3
9.1.2.3	Administrative Authority	9-4
9.2	CPF-Related Control Options	9-5
9.3	CA-Top Secret Command Keywords Used With CPF	9-7
9.3.1	Using DEFNODES With a TSS Command	9-9
9.3.2	Administering an ACID's DEFNODES	9-9
9.3.3	What Happens When a Command is Issued	9-11
9.4	CPF Recovery File	9-12
9.5	Recovery and Accountability	9-13
9.5.1	Security Files Among Networked Machines	9-13
9.5.2	System Entry Validation and Password Propagation	9-14
9.5.3	Installation Exit	9-14
9.5.4	TSSCPR Utility	9-14
Chapter 10.	Protecting Facilities	10-1
10.1	Administering the Facility Matrix Table	10-3
10.1.1	How to Display the Facility Matrix Table's Contents	10-4
10.1.2	How to Display a Facility's Definition	10-4
10.1.3	How to Add a New Facility	10-4
10.1.4	How to Change a Facility's Definition	10-6
10.1.5	How to Change a Facility's Name	10-8
10.1.6	Default Values for Predefined Facilities	10-9
10.2	Securing CICS	10-17
10.2.1	Signon Security and Authorization Restrictions	10-17
10.2.2	Password Validation	10-18
10.2.3	Terminal Security	10-18
10.2.4	Program Security	10-18
10.2.5	Job Submission Validation	10-18
10.2.6	Security Administration	10-18
10.2.7	Security Key	10-19
10.2.8	Multiple Signons	10-19
10.3	Securing CA-IDMS	10-20
10.3.1	Signon Security and Authorization Restrictions	10-21
10.3.2	Transaction Security	10-21
10.3.3	Password Validation	10-21
10.3.4	Program and Subschema Security	10-22
10.3.5	Area Security	10-22

10.3.6	Terminal Security	10-22
10.3.7	Security Administration	10-22
10.4	Securing BATCH Jobs	10-23
10.4.1	Signon Security and Authorization Restrictions	10-23
10.4.2	Password Validation	10-24
10.4.3	Card and Remote Reader Security	10-24
10.4.4	Job Submission Validation	10-25
10.4.5	Job Scheduling Security	10-25
10.5	Securing ICCF	10-26
10.6	Securing VM	10-27
10.6.1	How to Activate/Deactivate the VM Facility	10-28
10.6.2	Sharing Security Files Between VSE, VM and MVS	10-28
Chapter 11.	Customizing and Extending Security	11-1
11.1	Extending Security Using the Application Interface	11-2
11.1.1	How to Use the Application Interface	11-3
11.1.1.1	Request Record Field Characteristics	11-4
11.1.1.2	CA-Top Secret Return Data Fields	11-5
11.1.2	Performing Checks	11-8
11.1.2.1	General Resource Checking	11-9
11.1.2.2	Data Set and Volume Checking	11-10
11.1.2.3	Transaction/Command Checking	11-11
11.1.2.4	Facility Checking	11-11
11.1.2.5	Facility List Checking	11-11
11.1.2.6	Resource List Checking	11-12
11.1.2.7	Logging User Information	11-13
11.1.2.8	Dynamic Update Facility	11-13
11.1.2.9	Field Extract and Update Facility	11-15
11.1.2.10	Retrieving an ACID	11-16
11.1.2.11	Other Checks for the ACID	11-16
11.1.3	Programming Examples	11-17
11.1.3.1	Assembler	11-17
11.1.3.2	PL/I	11-18
11.1.3.3	COBOL	11-19
11.2	Extending Security Through the VSE Security Interface	11-21
11.2.1	The RACROUTE Macro Calls	11-22
11.2.1.1	The RACROUTE REQUEST=VERIFY Call	11-22
11.2.1.2	The RACROUTE REQUEST=AUTH Call	11-23
11.2.1.3	Dynamic Extract/Update Facility	11-26
11.2.1.4	Tips for Using DUF	11-29
11.2.1.5	The RACROUTE REQUEST=FASTAUTH Call	11-29
11.2.1.6	Class Name Formats	11-30
11.2.1.7	Specifying a Character String Class Name	11-30
11.2.1.8	Performance Shortcuts	11-33
11.2.1.9	Return Codes	11-33
11.2.1.10	Sample RACROUTE REQUEST=FASTAUTH Specification	11-34
11.2.1.11	The RACROUTE REQUEST=EXTRACT Call	11-34
11.2.2	Information Feedback	11-36
11.2.3	Multiple User Address Space	11-38
11.2.3.1	The CA-Top Secret Environment	11-39

11.2.3.2	Programming Examples	11-41
11.2.3.3	Data Areas Used by Sample Programs	11-44
11.2.4	Resource Name List Service Support	11-45
11.2.4.1	Invoking Support	11-45
11.2.4.2	Returned Format	11-46
11.2.4.3	Return Codes	11-46
11.3	Extending Security Through Site Security Exits	11-47
11.3.1	Customization Ideas	11-48
11.3.2	Mechanics	11-48
11.3.2.1	Common Exit Parameters	11-48
11.3.2.2	PREINIT	11-49
11.3.2.3	VOLUME	11-50
11.3.2.4	DATASET	11-50
11.3.2.5	RESOURCE	11-51
11.3.2.6	COMMAND	11-51
11.3.2.7	TERM	11-53
11.3.2.8	POSTINIT	11-53
11.3.2.9	UNDEFIND	11-54
11.3.2.10	PASSWORD	11-54
11.3.2.11	IO	11-54
11.3.2.12	SESSEND	11-55
11.3.2.13	SUBMIT	11-55
11.3.2.14	CHANGE	11-56
11.3.2.15	ACTION	11-57
11.3.2.16	MESSAGE	11-57
11.3.2.17	VIOLATN	11-58
11.3.2.18	SITE	11-59
11.3.2.19	CPF	11-59
11.3.2.20	TSSINSTX Characteristics	11-60
11.3.3	Activating the Installation Exit	11-61
11.3.4	CA-Top Secret Exit/User Entry Points	11-62
Appendix A. Computer Operations Guide		A-1
A.1	Control Options	A-3
A.1.1	Restricted Options	A-3
A.1.2	Authorization	A-4
A.1.3	CONSOLE Attribute	A-4
A.1.4	Entry Methods	A-4
A.1.5	Mode	A-5
A.1.6	BACKUP Control Option	A-5
A.1.7	BYPASS Control Option	A-6
A.1.8	CANCEL Control Option	A-7
A.1.9	DUMP Control Option	A-7
A.1.10	RECOVER Control Option	A-8
A.1.11	SECTRACE Control Option	A-8
A.1.12	STATUS Control Option	A-10
A.1.13	SUSPEND Control Option	A-10
A.1.14	SYNCH Control Option	A-10
A.1.15	SYSOUT Control Option	A-11
A.1.16	VERSION Control Option	A-11

A.1.17 TSS MODIFY Command	A-11
A.2 Startup Procedures	A-13
A.3 Shutdown Procedures	A-14
A.3.1 Temporary Shutdown	A-14
A.3.2 End-of-Day Shutdown	A-14
A.4 Restart Procedures	A-16
A.5 Security File Backup Procedures	A-17
A.5.1 CA-Top Secret Automatic Backup Feature	A-18
A.5.1.1 TSSBCKUP Procedure	A-19
A.5.1.2 Manual Backup of the Security File	A-19
A.6 Recovery and Audit/Tracking File Backup Procedures	A-20
A.6.1 Recovery File	A-20
A.6.2 Audit/Tracking File(s)	A-20
A.7 Security File Serialization	A-21
A.8 Messages and Codes	A-22
Appendix B. Prefixed Resources	B-1
Glossary	X-1
Index	X-7
User Registration Form	-URF-1
Demand Analysis Request Form	-DAR-1
Reader Comment Form	-RCF-1

About This Guide

Purpose

This guide is designed to explain to the security administrator what procedures need to be followed to administer security using CA-Top Secret. The information in this guide is separated into two parts.

- Part 1 provides an overview of the major capabilities, options, and features that CA-Top Secret provides. Basic concepts and terminology necessary for a fundamental understanding of CA-Top Secret are presented. Part 1 also contains a list of steps that should be followed when implementing CA-Top Secret for the first time.
- Part 2 contains general information that applies to all facilities when implementing CA-Top Secret. Information regarding functionality specific to a particular facility—such as BATCH or CICS—can be found in the *Implementation Guide* for that facility.

Also included in this guide is information about techniques for customizing CA-Top Secret to extend its security capabilities to satisfy specific site-dependent requirements. For facility specific customization techniques and capabilities refer to that facility's implementation guide.

In addition to security administrators, other users who may find the information in this guide helpful are:

- Personnel responsible for the implementation and maintenance of CA-Top Secret, and
- Systems programmers involved in implementing basic CA-Top Secret capabilities and customizing additional capabilities.

Organization

PART 1: CA-Top Secret Fundamentals

Chapter	Description
1	Explains basic concepts and introduces CA-Top Secret terminology.
2	Lists and describes implementation steps, including instructions on implementing phased security.
3	Describes CA-Top Secret violation, logging, reporting, and auditing capabilities.

PART 2: Using CA-Top Secret to Develop Your Security Environment

Chapter	Description
4	Contains explicit instructions for creating security administrators, user ACIDs, profiles, and groups. Also explains how to display information and use control options.
5	Explains password and system entry security.
6	Illustrates the basic procedures for assigning resource ownership and controlling access to resources. The operation of the Security Validation Algorithm is also explained.
7	Provides detailed instructions for protecting resources.
8	Describes how to maintain special security records, like the Resource Descriptor Table, the Field Description Table, and the ALL record.
9	Examines what the Command Propagation Facility is and how to use it.
10	Gives a general overview of the capabilities that can be used to secure each facility, and explains how to administer the Facility Matrix Table.
11	Details how to customize and extend security using the Application Interface, the VSE Security Interface, and site security exits.

Chapter	Description
Appendix A	Outlines the computer operator's functions and responsibilities.
Appendix B	Provides a list of prefixed resources.

Chapter	Description
Glossary	Provides easy-to-understand definitions for many CA-Top Secret and mainframe terms.
Index	Provides an efficient way to locate specific material.

CA-Top Secret Publications

The following publications are supplied with CA-Top Secret:

Title	Contents
<i>Installation Guide</i>	CA-Top Secret installation for VSE, including: installation and maintenance steps, startup and shutdown considerations, backup and recovery procedures, and Security File conversion.
<i>Command Functions Guide</i>	Comprehensive reference to the TSS command and CA-Top Secret resources.
<i>Control Options Guide</i>	Comprehensive reference of CA-Top Secret control options.
<i>Implementation: BATCH, STC and APPC Guide</i>	Provides for setting up security in Batch, STC and APPC environments.
<i>Implementation: CICS Guide</i>	Provides for setting up security in a CICS environment.
<i>Messages and Codes Guide</i>	Provides all CA-Top Secret messages and codes.
<i>Planning Guide</i>	General guide for planning a successful CA-Top Secret security implementation and describing the latest enhancements to CA-Top Secret.
<i>Report and Tracking Guide</i>	Details the use of TSSUTIL, TSSTRACK, TSSAUDIT, TSSCFE and TSSCPR and provides sample reports.
<i>Troubleshooting Guide</i>	Includes CA-Top Secret debugging options and facilities, information to be prepared prior to contacting Technical Support, and an explanation of customer support.
<i>User Guide</i>	Conceptual overview of CA-Top Secret architecture and capabilities. Also includes implementation instructions that span all facilities, including: implementation how to's, customization, and the CA-Top Secret utilities.

All guides are updated as required. Instructions accompany each update package.

Related Publications

The common component services formerly known as CA90s have been enhanced and renamed CA Common Infrastructure Services, CA-CIS. All the documentation for the services exists in the following publications supplied by Computer Associates:

Name	Contents
CA-CIS Administrator Guide	A guide for system administrators.
CA-CIS CAICCI User Guide	Describes how to use the communication protocols needed for cross-system communication.
CA-CIS Installation Guide	Tells how to install the CA-CIS.
CA-CIS Message Guide	Lists messages and codes for CA-CIS.
CA-CIS Reference Guide	Describes how to access and use the VSE common components.
CA-CIS Systems Programmer Guide	Details the programming requirements for command, security, and signon exits.

The following publications relate to CA-Top Secret and are available from Computer Associates:

Title	Operating System
<i>CA-EARL Reference Guide</i>	MVS/VM/VSE
<i>CA-EARL User Guide</i>	MVS/VM/VSE
<i>CA-EARL Systems Programmer Guide</i>	VSE
<i>CA-EARL Examples Guide</i>	MVS/VM/VSE

Command Notation

Enter the following exactly as they appear in command descriptions:

UPPERCASE	identifies commands, keywords, and keyword values which must be coded exactly as shown.
MIXEDcase	identifies acceptable command abbreviations. The UPPER-CASE letters represent the minimum abbreviation possible. The lowercase letters of the command may be entered for further clarification Note: In some cases, the command processor may require a more detailed abbreviation due to user-defined resource being the same as a command abbreviation. In these cases, either enter the complete command or code a national or numeric character in one of the first four positions of the user-defined resource.
<u>underlining</u>	identifies default operands. These operands do not need to be entered to take effect.
Symbols	"" / * # () , must be coded exactly as shown.

The following items clarify command syntax; do not type when they appear:

lowercase	indicates keyword values which you must supply.
[]	identifies optional keywords.
	separates alternative keywords or values; choose one.
{ }	indicates that you must enter one of the keywords.
...	means the preceding value may be repeated more than once.

The following table indicates the appropriate command format.

Sample Command	Explanation
TSS PER(acid) DSN(dsname)	You must supply a value for the ACID and for the data set name.
MODE(DORM IMPL WARN FAIL)	You must choose only one of the keywords.
TSS {ADDto } (acid) MCSAUTH {(INFO) } {REMove } {(MASTER)} {REPlace} {(SYS) } {(IO) } {(CONS) } {(ALL) }	You must select a command function, enter the name of an ACID, enter the MCSAUTH keyword, and select an operand from the list.

Chapter 1. Introducing CA-Top Secret

1.1 Why Use CA-Top Secret?

CA-Top Secret is an important component in the Computer Associates family of integrated security solutions, designed to streamline security administration, enable single-point user signon, and provide auditing capabilities.

CA-Top Secret offers a unique hierarchical security configuration similar to the organizational structure of your corporate environment. Administrative tools, reporting options and automatic logging capabilities help ensure the integrity and security of your corporate assets.

1.1.1 Purpose

The purpose of security software is to minimize the risks of accidental or intentional corruption, destruction, or disclosure of data. The integrity of the information kept in the computing environment is essential. CA-Top Secret provides that integrity.

Through individual accountability, access permissions, and a comprehensive audit trail, CA-Top Secret controls and monitors who can access and change data. Until permissions are issued to control how new data is shared, that data is protected by default.

Additional advanced technology within CA-Top Secret provides further assurance of data integrity (such as, the capability to contain viruses from spreading uncontrolled through production programs). Also, CA-Top Secret is fully SAF-compliant—providing controlled mechanisms for access to security information.

1.1.2 Components and Features

The following CA-Top Secret components and features ensure your environment's integrity and security:

- Encompasses BSM Functions
- System Entry Validation
- Resource Protection
- User Information Repository
- Distributed Security
- Enhancement to Security Interfaces

Encompasses BSM Components: As with BSM, CA-Top Secret provides system entry validation, resource control, auditability, accountability, administrative control, and SAF compatibility.

Unlike BSM, CA-Top Secret has a user-oriented architecture, which means that the focus of security is on the individual user rather than on the resource. This type of architecture offers less overhead, more efficiency, and better overall implementation.

System Entry Validation: CA-Top Secret requires that the user have a valid ACID (ACcessor ID) and password before entering the system. Other system entry validations are optional and can be tailored to meet your environment. For example, you can restrict a particular user by entering the system through a specific facility, such as TSO.

Resource Protection: A resource is any component of the computing or operating system required by a task. CA-Top Secret protects a wide variety of resources by default, and provides the capability of also protecting site-defined resources.

User Information Repository: As part of building your security database, you must identify each user to CA-Top Secret by using ACIDs. The attributes and resource access permissions you associate with each ACID comprise the ACID's Security Record which, in turn, becomes part of the overall Security File.

Security database, Security File, and Security Record are discussed in the section *How Does CA-Top Secret Work?* later in this chapter.

CA Common Infrastructure Services Architecture: The CA Common Infrastructure Services (CA-CIS), formerly known as CA90s, enables CA to deliver Systems Management Software solutions that perform better, are more deeply integrated, exhibit greater reliability, and are consistent across hardware platforms. These benefits enable you to make significant strides in achieving total data center automation, in meeting service-level requirements, and in controlling costs while maximizing the return on investment.

Distributed Security: The security area is a good example of the cross-platform capability of CA-CIS. Distributed security in CA-Top Secret coordinates multi-system security by using the following components of the Integration Services of CA-CIS architecture:

- **CAIENF** (Event Notification Facility) is an operating system interface component that offers a simple, yet flexible, approach for CA-Top Secret to obtain data from VSE. CAICCI provides the VTAM facilities needed to transmit and receive TSS commands, when using the Command Propagation Facility (CPF).
- **CAISSF** (Standard Security Facility) provides an easy-to-use application interface for CA and non-CA products to obtain and use CA-Top Secret security information.
- **CAICCI** (Common Communication Interface) is a common communications facility that enables CA-Top Secret secured nodes to communicate with one another.

Within CA-Top Secret, distributed security can include the following:

- **Command Propagation Facility (CPF)** is a major part of distributed security in CA-Top Secret that can route security administration to all or selected nodes either synchronously or asynchronously, resulting in single point administration. Changes made to ACIDs, passwords, or access levels, for example, can be propagated to all nodes to which the user is defined. For example, USER01 is defined to two nodes, with NODE A as his local node and NODE B as his remote node. If he changes his password on NODE A, CPF will automatically propagate the change to NODE B. Through the use of command function keywords, you can specify which node receives these commands and how the local node processes them.

For more details on CPF, refer to the section *Distributed Security Processing* later in this chapter.

Enhancement to Security Interfaces:

CA-Top Secret offers security interfaces not only for CA products, but for many OEM products as well. The *Implementation: Other Interfaces Guide* contains information about installing and implementing CA-Top Secret when any of these products are being used.

1.2 How Does CA-Top Secret Work?

The purpose of CA-Top Secret is to control access to resources by limiting system entry (through passwords and terminal restrictions, for example) and limiting how, when, and which resources a user can access once he enters the system.

As part of using CA-Top Secret to secure your installation, the security administrator must design the security database. The following terms and their distinctions are important to understand the basics of CA-Top Secret:

Security Administrator	The person who is primarily responsible for implementing and maintaining system security by defining users, resources, access levels, and facilities.
Security Database	A systemized collection of data—containing information on user and resource definitions, as well as access permissions and system entry conditions stored for immediate use.
Security File	An encrypted security database consisting of the Security Records, which contain all user and resource permissions and restrictions.
Security Record	A part of the Security File that contains a set of user and profile records, including such information as which resources a user can access, and how he can use them.

In addition, the security administrator is also responsible for:

- Controlling the security environment through the use of control options and TSS commands.
 - Control options provide a mechanism allowing CA-Top Secret to customize the environment.
 - Control options are typically set during installation, and are stored in what is called the Parameter File.
 - One of the most important control options is MODE, which determines how CA-Top Secret will react to a particular resource access request or violation.
 - TSS commands are used primarily to define ACIDs and to establish resource access. Many control options can be temporarily modified using the TSS command.
- Monitoring resource access violations through the CA-Top Secret auditing, tracking, and reporting options.

CA-Top Secret also provides the means for securing subsystems and facilities (such as CICS, VM, CA-VOLLIE, and CA-IDMS), as well as for maintaining security across multiple VTAM-connected nodes (through the Command Propagation Facility).

1.2.1 System Entry Protection

Once an ACID has been defined to CA-Top Secret, the user must pass at least two "tests" to enter the system.

1. He must have a valid ACID.
2. He must have a valid password.

As part of a flexible approach to security administration, CA-Top Secret supports a wide variety of password security policies by using password protection controls and options. Some of these controls and options are:

- Restricting how passwords can be defined and changed
- Setting expiration intervals
- Maintaining password history
- Selecting a password violation threshold

In addition to these password controls, an ACID can also be restricted to:

- A particular terminal or CPU
- Access only on particular days of the week or during certain hours
- Access through particular facility such as CICS.

1.2.1.1 Types of ACIDs and How They're Structured

ACIDs are the ACcessor IDs by which users are identified to CA-Top Secret. An ACID can be up to eight alphanumeric characters long, which normally corresponds with the user's system userid. With CA-Top Secret, you have the option of using the same ACID for all facilities or using a different ACID for each facility (that is, CICS, VM, and so on).

CA-Top Secret recognizes several different types of ACIDs, ranging from a user to an entire zone. Together, these types comprise the basic hierarchical structure of your CA-Top Secret security database. Each of these ACID types is then associated with a set of resource access authorizations.

The simplest, and generally, the most useful way to build your security database is to create a security hierarchy that mirrors your actual corporate structure. However, CA-Top Secret does not limit you to this approach. The basic function of the corporate structure defined to CA-Top Secret is to serve as a coordinating framework for security administration at your installation. Therefore, its design should be fundamentally dictated by what will best facilitate security implementation and maintenance.

The CA-Top Secret hierarchy is constructed from seven ACID types:

- User
- Profile
- Group
- Department
- Division
- Zone

- Control

These ACID types fall into one of two categories:

- Functional, or
- Organizational

Functional ACIDs refer to User, Profile, Group, and Control ACIDs, and are used to perform specific tasks. Organizational ACIDs are Department, Division, and Zone ACIDs, and are used to construct the upper levels of your security hierarchy. Functional ACIDs "report to" organizational ACIDs, while organizational ACIDs "report to" other organizational ACIDs (organizational ACIDs never "report to" functional ACIDs).

The maximum size of an ACID is 256K.

Let's take a closer look at the different ACID types and their role within the CA-Top Secret hierarchy.

User ACIDs: At times, the difference between a user and a User ACID can be confusing. Just remember that a user is a person. A User ACID designates a specific employee in a department—the lowest level of the CA-Top Secret organizational hierarchy—but can refer to any ACID type. Individuals are associated with User ACIDs or Control ACIDs.

Every User ACID must be associated with a single Department ACID.

Profile ACIDs: When a group of users need to use a set of identical resources in the same way (the users perform similar or related job functions), it is convenient to define this set of access authorizations once and then associate the entire set with each of the users in the group. In CA-Top Secret this set of common resource access characteristics is termed a *Profile*. Every profile is assigned a unique Profile ACID.

Once a profile is defined it can be associated with any number of users (at the same or different levels in the hierarchy), thereby eliminating the need to define each resource access authorization separately for every user.

Every Profile ACID must be associated with and defined to a single Department ACID.

Group ACIDs: CA-Top Secret supports the concept of Groups in the IBM OpenMVS environment. A *Group* is similar to a Profile in that it is a collection of users who can share access authorities for protected resources.

Department ACIDs: At any installation, users typically work for a particular department. CA-Top Secret recognizes this logical separation by requiring each User ACID to be associated with one Department ACID.

Division ACIDs: CA-Top Secret lets you optionally define multiple divisions within your corporate security structure. Each division can be composed of one or more departments. (A department doesn't have to be associated with a division). Every division is

assigned a unique Division ACID, and resources can be assigned to a Division just as they can be assigned to a Department, Profile, or User.

Zone ACIDs: A *Zone* ACID is another optional organizational level in your corporate security structure. A zone can be used to group two or more divisions. Every Zone is assigned a unique Zone ACID and—as with Users, Profiles, Departments, and Divisions—resources can be assigned to a zone as well.

Control ACID: Control ACIDs are used for administrative purposes and define security administrators that are associated with various structural levels within the CA-Top Secret security database. A *Control* ACID, like an ordinary User ACID, can be a regular user of system facilities. A Control ACID can issue subsystem commands and perform other functions—such as access data sets and submit jobs.

Initially, CA-Top Secret knows of only one Security Administrator—the MSCA ACID. This ACID is defined to CA-Top Secret during the installation process; other Control ACIDs are created later.

Each type of Control ACID performs administrative tasks for the structural level it is associated with. To enable the Control ACID to perform these tasks, each one is assigned a scope of authority and administrative authorities within that scope.

Concept Of Scope: Once you define your security administrator ACIDs, you need to consider for which part of your security hierarchy they will be responsible. This is called the security administrator's scope of authority. CA-Top Secret provides you with several different levels of Control ACID scope, and each level corresponds to a level in your corporate structure. For example, a Division Control ACID, or VCA, is responsible for administering security for all the ACIDs within a particular Division (including ACIDs assigned to Departments associated with that Division). Referring to Figure 1-1 on page 1-9, the VCA responsible for the Finance Division would also be responsible for the ACIDs within the Payroll and Accounting Departments.

Concept Of Authority: In addition to scope of authority, the Security Administrator must also be assigned particular types of administrative authorities. These authorities define the security functions the Control ACIDs can perform for ACIDs within their scope.

Upper level security administrators can grant administrative authorities to lower level administrators within their scope, provided the higher level administrators already possess the appropriate authorities.

Figure 1-1 on page 1-9 illustrates a typical corporate structure and the ACIDs related to each structural element.

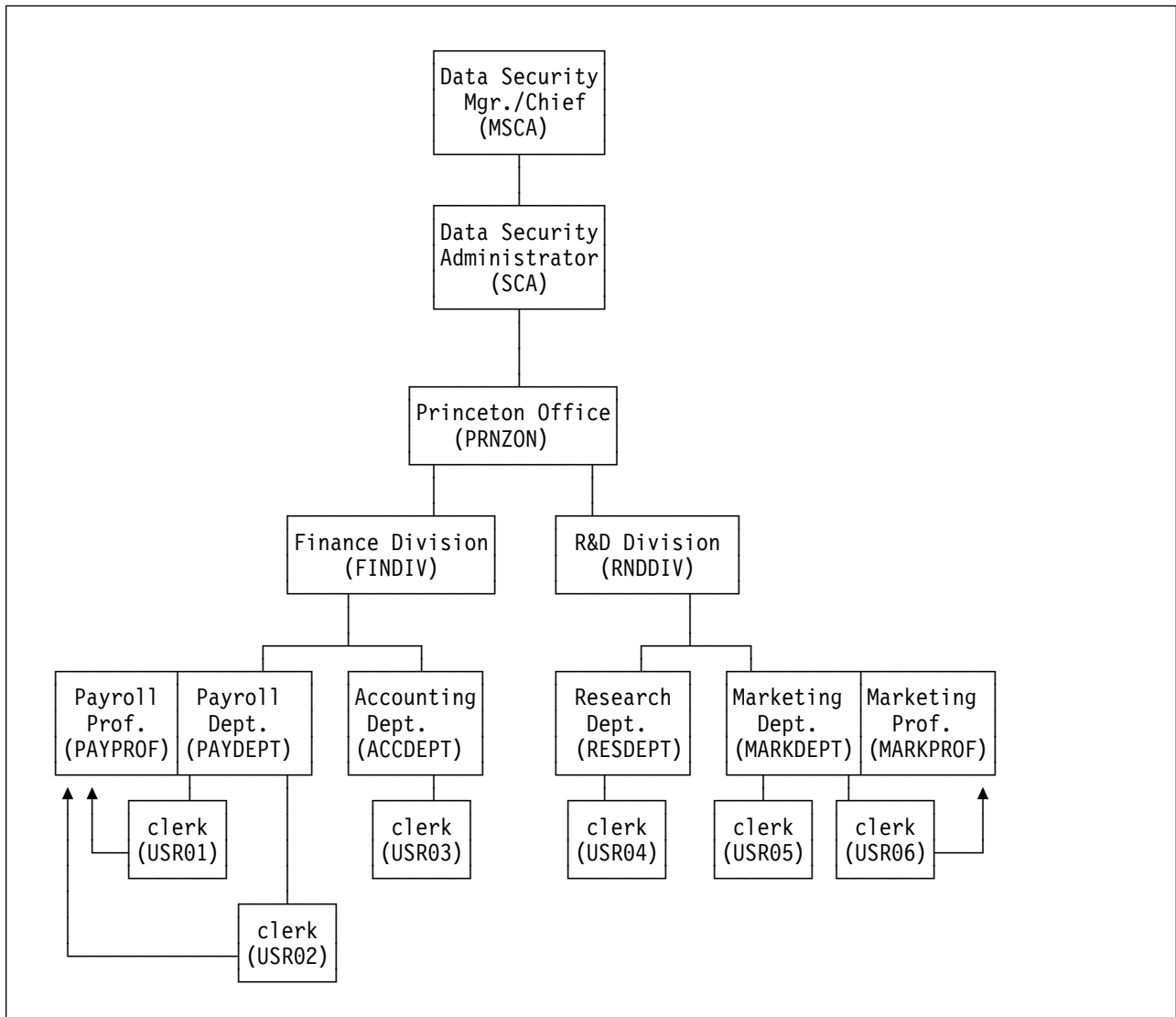


Figure 1-1. Corporate Hierarchy

The following table illustrates how the CA-Top Secret ACIDs correspond to elements within the corporate structure shown in Figure 1-1 on page 1-9.

Table 1-1. How ACIDs Correspond to Structural Elements	
Corporate Element	Corresponding ACID
Data Security Mgr./Chief	MSCA is a Control ACID
Data security administrator	SCA is a Control ACID
Princeton Office	PRNZON is a Zone ACID
Finance Division R & D Division	xxxDIV are Division ACIDs
Payroll Department Accounting Department Research Department Marketing Department	xxxDEPT are Department ACIDs
Payroll functions Accounts Receivable functions Accounts Payable functions	xxxPROF are Profile ACIDs
Clerks	USRxx are User ACIDs

1.2.1.2 The Role of the Security Administrator

The security administrator (or security administrators, depending on your ACID hierarchy) is the focal point of security for your site. He needs to understand how CA-Top Secret works and how to best implement security for your system. He may not necessarily be responsible for installing or maintaining the CA-Top Secret product, but he acts as the liaison between CA-Top Secret security and the users who need to access the facilities and resources it secures. The security administrator's actual role is determined by a number of factors, among them:

- His scope of authority (is he responsible for the entire installation or a single department within that installation).
- His designated administrative authorities (can he create ACIDs, run reports, change control options).
- Security needs of the site.

For example, you might design one security administrator whose sole purpose is ACID creation and maintenance, and another security administrator who is responsible for maintaining resources and the Resource Descriptor Table (RDT). For more details on the RDT, refer to Chapter 8, "Maintaining Special Security Records" on page 8-1. Or, like many installations, you might create a "backup MSCA" by assigning full authority to one SCA.

Regardless of his specific role or area of control, each security administrator must first possess the appropriate administrative authority.

Types of Administrators: The CA-Top Secret administrative hierarchy has seven levels:

- Master Security Control ACID (MSCA)
- Central Security Control ACID (SCA)
- Limit Central Security Control (LSCA)
- Zonal Control Acid (ZCA)
- Divisional Control ACID (VCA)
- Departmental Control ACID (DCA)
- User ACID

The first six levels represent types of ACIDs—that is, ACIDs whose primary function is to control security administration. (The MSCA is frequently referred to as the Master Central security administrator, the SCA as a Central security administrator, and so on.) While users can be given administrative authority (limited to themselves), their primary function is to perform work (for example, production processing). Likewise, Control ACIDs can perform work, but this should be a secondary function for them.

Types of Administrative Authorities: An ACID's authority determines what he can do with respect to the administration of ACIDs, resources, facilities, the displaying of security database information, and so on. An administrator can confer only those administrative authorities she already possesses herself.

The different types of administrative authorities are:

- ACID
- DATA
- RESOURCE
- FACILITY
- MISC1
- MISC2
- MISC8
- MISC9
- SCOPE

Each of these types of authority roughly corresponds to a different set of security environment control and maintenance functions (for example, ACID maintenance or resource maintenance). In addition, a group of operands is associated with each type of authority. Each operand designates a very specific functional authority. For example, ACID(CREATE) authority allows the Control ACID to create and delete ACIDs within her scope, while RESOURCE(INFO) allows her to perform certain inquiries for any resource within her scope.

Administrative authorities can't be assigned to a Division, Department, or Profile ACID.

Establishing Global Authorities: To give every ACID the ability to perform specified administrative functions, the administrator can assign the administrative authority to the ALL Record. For example, assigning MISC1(LTIME) to the ALL Record gives every ACID the authority to set his own terminal lock time interval. The ALL Record can also contain resources access levels.

1.2.1.3 Passwords

Password use and validation are the most fundamental mechanisms for protecting ACIDs from unauthorized use. CA-Top Secret requires that all ACIDs be password protected by default. A security administrator assigns the first password. The user associated with the ACID can then either change the password immediately, or later when it expires.

Password assignment is controlled by certain CA-Top Secret control option values. These values are set and stored within CA-Top Secret; however, they can be changed to fit your site's security requirements at any time.

Password assignment can be controlled in these ways:

- By following rules for changing passwords
- By setting password expiration intervals
- By using password violation thresholds
- By using random password generation

Each method is briefly described next.

Rules for Changing Passwords: The following rules demonstrate some of the options you can apply to all users of CA-Top Secret security:

- The minimum length of a password can be set. New passwords must be at least four characters in length and can't be exactly the same as, or even a close variant of, the user's previous password (for instance, "ninth" and "tenth" could be considered close variants because they differ by only two letters).
- Passwords can only be changed after a specific interval of time has elapsed (the system default is one day).
- Passwords can't contain repeating characters and can be required to conform to a mask.
- Passwords that match the userid or the first four characters of any word in the associated (personal) name field, as well as passwords that match entries in a restricted password list, are not allowed.

Password Expiration Intervals: A password expiration interval is the number of days before CA-Top Secret forces a user to change his password.

Password Violation: To prevent unauthorized system entry by "password guessers", CA-Top Secret recognizes a password violation threshold that is enforced system-wide. Once this threshold is exceeded, CA-Top Secret automatically suspends the ACID.

Random Generation: Random password generation is a feature that lets CA-Top Secret automatically generate a *random* set of characters for a password. A security administrator can instruct CA-Top Secret to generate a random password for a user whose password has expired, or the user himself can instruct CA-Top Secret to automatically generate a password.

1.2.1.4 ACID and Password Validation

Because of their importance in the CA-Top Secret security environment, ACIDs are validated in many different ways to protect against unauthorized use. First, CA-Top Secret checks the Security File to determine whether a designated ACID has been defined (by seeing if a Security Record exists for it). Second, if the ACID is undefined, CA-Top Secret responds based on the initial control option settings for the security mode and various system options.

Password validation occurs when the user signs on, supplying his ACID and its associated password. CA-Top Secret verifies that the supplied password is correct by checking it against the ACID/password combination stored within CA-Top Secret. If the supplied ACID/password combination matches, the user is allowed to continue; if it doesn't match, the user is denied access to the system. By combining password assignment and validation in this way, CA-Top Secret lets you secure your environment by controlling who can access it.

1.2.1.5 Modes

Mode is the level of security under which CA-Top Secret operates. There are four modes: DORMANT, WARN, IMPL, and FAIL.

These security modes are an invaluable tool when planning a phased or gradual implementation of your security environment. They can be set to apply to the entire installation, a particular facility, or even a particular user or group of users. Each mode is briefly introduced below and discussed in detail in Chapter 2.

Note: Passwords are validated in all modes.

DORMANT: CA-Top Secret is installed, but isn't actively validating. The TSS command is fully supported and can be used.

WARN: CA-Top Secret is active, but violations generate warning messages rather than the requests being failed.

IMPL: CA-Top Secret is active, and fails any unauthorized access requests. Users not defined to CA-Top Secret can operate normally but are restricted from accessing protected resources.

FAIL: CA-Top Secret is in full control of access requests. All users must be defined and all resources protected.

1.2.2 What is a Facility?

A facility is a way of grouping options and associating them with a particular service that users sign on to. BATCH is an example of a facility. CA-Top Secret provides security for many facilities, including:

- VM
- CICS
- CA-IDMS
- BATCH

As delivered, CA-Top Secret comes with many facilities already defined in what is known as the Facilities Matrix Table. You can customize this table by adding facilities and changing existing ones.

1.2.3 What is a Resource?

As was discussed previously, a resource is any component of the computing or operating system required by a task. CA-Top Secret protects a wide variety of computer resources, but to protect them it must know about them. Computer resources are secured through ownership and authorization.

Types of Resources: The types of resources (such as data sets, volumes, terminals and minidisks) that CA-Top Secret protects are listed in what is called the Resource Descriptor Table (RDT). Many resource types are already automatically defined to the RDT at installation; however, additional resource types (including site-defined resources) can be added. A complete list of what is included in the RDT can be found in the *Command Functions Guide*.

Ownership And Authorization: Securing resources is a two step process. Once the resource type, or *class*, is defined in the RDT, then each resource must be:

- owned by an individual or department ACID
- permitted to additional ACIDs (if necessary)

Ownership of a resource automatically implies full access to that resource. For other ACIDs to have access to that resource, they must be *authorized*, or permitted, to use it.

Security Validation Algorithm: Once all resources have been defined to CA-Top Secret and their access levels specified, any future request to access those resources is processed through the CA-Top Secret *Security Validation Algorithm*. The Security Validation Algorithm is the formula that CA-Top Secret uses to determine whether an ACID has the appropriate authorizations to access a particular resource.

The Security Validation Algorithm is discussed in detail in Chapter 6.

Relationship Between Data Sets And Volumes: When an ACID requests access to a particular DASD data set, CA-Top Secret must potentially evaluate both the pertinent volume and data set authorizations—depending on what is known as bypass options associated with that ACID or its authorization.

In the event that both volume and data set level access checking is being done, CA-Top Secret always performs volume level first. In some cases, a request to access a data set is granted or failed strictly on the basis of the ACID's volume access authorizations.

1.2.4 What is a Field?

A field resides in the data area of the ACID's Security Record which holds information to be used by system and security level applications.

Security Record Segments and Fields: Each Security Record is broken down by segments as defined in the *IBM External Security Interface (RACROUTE) Guide*.

Fields and segments of the ACID Record are defined in a special Reserved Record of the Security File known as the Field Descriptor Table (FDT). Some fields and segments are pre-defined by CA-Top Secret at installation; however, additional fields and segments can be added. For more details, refer to the *Command Functions Guide*.

Administrative Authority: Manipulation of the Field Descriptor Table (FDT) requires special administrative authority.

Assigning Values for Defined Fields: After fields and segments are defined to the FDT, the field name becomes a keyword in CA-Top Secret. Values are assigned, replaced and removed using standard CA-Top Secret commands. Fields in an ACID Security Record can be modified by anyone with administrative authority over the ACID.

1.2.5 Control Options and Command Functions

Control options and command functions are used to communicate with CA-Top Secret. The basic distinction between control options and command functions is that control options define your security environment and command functions are used to maintain the integrity of the security database.

1.2.5.1 What are Control Options?

Control options are used to customize the security environment of a particular installation. Control options are typically set during installation, and are stored in the Parameter File. One of the most important control options is MODE, which determines how CA-Top Secret reacts to a particular resource access request or violation. Many control options can be temporarily changed using the TSS MODIFY command function.

1.2 How Does CA-Top Secret Work?

The following example tells CA-Top Secret to modify the FACILITY control option so that users on the BATCH facility will be in IMPL mode.

```
TSS MODIFY(FAC(BATCH=MODE=IMPL))
```

Control options are discussed in detail in Chapter 4.

1.2.5.2 What are Command Functions?

Command functions are the primary tool of the security administrator and are always preceded by the letters TSS. A command function is used to define ACIDs, assign attributes, and determine resource access.

For example, the following TSS command will assign a specific date on which an ACID will expire.

```
TSS ADD(USER01) UNTIL(11/24/96)
```

All command syntax components are described in the examples on the next page.

```

1 2           3           4 5
TSS FUNCTION { (acid) } KEYWORD(OPERAND)
                { (ACIDS) }
                { (STC) }
                { (AUDIT) }
                { (RDT) }
                { (FDT) }
                { (DLF) }
                { (ALL) }
    
```

Table 1-2. TSS Command Syntax		
Component	Description	Rules
1	TSS command name.	Command must always begin with TSS.
2	Name of the function CA-Top Secret will perform.	1. Must immediately follow TSS. 2. Only one function entered per TSS command. 3. One or more spaces must be entered between TSS and the function.
3	Specifies the ACID being affected by the function.	ACID names can be up to eight characters long and must conform to the restrictions established by your site.
4	Specifies the resource type or security attribute being processed by the function.	1. Keywords can be entered in any order. 2. Online: Keywords can be entered from line to line without special action. 3. Batch: The last keyword on a continuing line must be followed by a blank and a dash. The next keyword can be entered on the next input line.
5	Enter the specific prefix, resource name, or the required value or 5 name for a security attribute.	Operands must be provided and parentheses are required to indicate no value. If an operand is missing, any following keyword is ignored.

1.2.5.3 Entry Methods

CA-Top Secret functions can be entered freeform onto the command screen of an online terminal.

Freeform: The following screen illustrates how commands are entered freeform onto the command screen:

```
TSS CREATE(USER01) TYPE(USER) NAME('H.PARKER') PASSWORD(1234,30,EXPIRE)
SOURCE(GRAF0076) PROFILE(BUDGET,TAXES,CRIME) DSN(SYS.01)
DEPT(DEPTB01)
```

1.2.6 Distributed Security Processing

As discussed earlier, distributed security processing can include:

- Command Propagation Facility (CPF)
- CA-Top Secret/VM
- CA-Top Secret/MVS
- Unicenter Workstation for CA-Top Secret Security Management

This section focuses on the major component of distributed security processing—the Command Propagation Facility (CPF).

1.2.6.1 What is the Command Propagation Facility (CPF)?

With the Command Propagation Facility, distributed security processing allows you to administer security across multiple VTAM nodes. For example, with the appropriate authorization a security administrator on one node can make modifications to the Security File on another node. The Command Propagation Facility allows centralized control of the whole network or even a smaller portion of that network.

The Command Propagation Facility provides the security environment with:

- Routing of security administration to all or selected nodes within the MVS, VM, or VSE security network.
- Optional synchronous or asynchronous remote command execution. The basic difference between the two types of command execution is that synchronous waits for the command response to return from the remote node before continuing, while asynchronous doesn't have to wait for a response before resuming processing.
- TSS command execution with most CA-Top Secret commands.
- Automatic update of passwords on all connected systems if changed by the user at logon.
- Propagation of user-initiated suspensions for exceeding password and violation threshold limits.

- Optional Journal Files (SYSLST, tape, or disk) to log commands transmitted to, and responses received from, remote nodes.
- Optional collection of asynchronous commands in a Recovery File so that they can be retransmitted in case of network outage.

1.2.6.2 Synchronizing Information Across Nodes

CPF allows you to automatically synchronize Security Administration on multiple nodes through the propagation of TSS commands, as well as user-initiated changes—such as suspension and password changes. Security administration propagation can be either implicit or explicit. Implicit uses the CPF control options to set system-wide propagation rules; while explicit uses CPF command keywords to set propagation rules on a command-by-command basis.

1.2.6.3 Controlling Access From Remote Nodes

When CPF transmits a command to a remote destination, it records the command image on the Journal File for that node and associates an ID with that command. A Journal File provides an historical record of the command traffic to and from CA-Top Secret. When a response is received from the remote node, CPF journals the response and the ID number so that the response can be matched to the command that prompted it. When the response is sent back, it is journalled with the ID and remote destination name. By examining the appropriate Journal File, an auditor can see exactly what came in, what went out, and the results of the action taken.

1.3 Where is Information Stored?

Information for CA-Top Secret is stored among many files and records which work together to provide an integrated security software package.

1.3.1 CA-Top Secret Files

The files used by CA-Top Secret to secure an environment are:

- Security File
- Parameter File
- Audit/Tracking File (optional alternate file)
- Backup File
- Recovery File
- Command Propagation Files

A brief introduction of each of these files appears next; a detailed discussion appears in Chapter 9, *Maintaining Special Security Records*.

1.3.1.1 Security File

This file is an encrypted security database consisting of the Security Records that contain all user and resource permissions and restrictions. When a user initiates a job or signs on to an online facility in a VSE environment, CA-Top Secret obtains the user's Security Record from the Security File, and places it in the user's address space for the duration of the session.

1.3.1.2 Parameter File

This file stores and defines control options at initialization, and sets up the operating environment for CA-Top Secret. As was discussed earlier, control options are the tools that allow you to modify and customize the security environment.

1.3.1.3 Facility Matrix Table

This table contains all the facilities defined to CA-Top Secret. Each entry contains information about the specific attributes associated with a particular facility (like VM, CICS, and so on), and can be viewed and modified with the FACILITY control option.

1.3.1.4 Audit/Tracking File

This file records security-related events and can be shared among CPUs. These events include violations, job and session initiation, and resource access. You can also designate an optional, alternate Audit/Tracking File to increase the amount of information that can be stored before the file fills up and begins to wrap.

1.3.1.5 Backup File

This file stores the automatic daily backup of the Security File to ensure complete integrity of the security environment. The backup file is an exact copy of the Security File, as it existed at the time of last backup, and can be used if the Security File device becomes unavailable.

1.3.1.6 Recovery File

This file is a wraparound file that stores recent administrative commands depending on the size of the file allocated. The backup Security File with the application of select recovery file commands can completely restore a damaged Security File.

1.3.1.7 \$\$LOG\$\$ File

CA-Top Secret will dynamically allocate a SYSLST output member. This member contains the following information:

- All modifications to the CA-Top Secret default control options, originating from:
 - CA-Top Secret Parameter File
 - Console Operator command modification of CA-Top Secret task
 - Online administrator TSS command modification
- Status messages for the CA-Top Secret task regarding:
 - Security File capacity and backup
 - Audit/Tracking File capacity and backup
 - I/O errors in CA-Top Secret files during task execution

1.3.1.8 Command Propagation Files

There are two distinct types of files that are associated with the Command Propagation Facility (CPF): the Recovery File and the Journal File. These files must be dedicated to a single CPU.

CPF Recovery File: This file is a disk file used by the Command Propagation Facility to save transmitted commands until a response to those commands has been received from remote nodes. There is one Receive journal to record commands and their response from other nodes. There can be a Send journal for each connected node to record commands and responses.

CPF Journal File(s): These files provide an historical record of the command traffic to and from a particular CA-Top Secret CPF node.

1.3.2 Special Security Records

CA-Top Secret has several reserved or special ACIDs that are pre-defined and maintain resource and attribute information. These records include:

ALL Record	Identifies resources that are globally accessible to all signed on users.
Audit Record	Stores the resource names that are to be audited.
APPCLU Record	Stores the names and security requirements of the logical units (LUs) involved in APPC conversations.
Resource Descriptor Table (RDT)	Contains pre-defined resource classes. Each resource class is identified by a unique keyword and has certain attributes associated with it.
Field Descriptor Table (FDT)	Defines fields (classes) that can be attached to ACIDs within the Security File. Each field description contains a field name, field code, and field attributes.
Static Data Table (SDT)	Contains record elements that can be used to control which users have access to certain resources.
Node Descriptor Table (NDT)	Contains all PassTicket application and session key-related node information. The NDT is a global record similar to the Resource Descriptor and Field Descriptor Tables.

These records are discussed in detail in Chapter 8, *Maintaining Special Security Records*.

Chapter 2. Implementing CA-Top Secret

Your CA-Top Secret implementation strategy depends on the size and complexity of your organization, and the users and resources that are most critical. This chapter provides guidelines for developing a workable implementation strategy regardless of the size or complexity of your organization.

The steps that are listed below discuss at length the strategies that are suggested in the *Planning Guide*. It is recommended that you follow the order that is given in the implementation steps, wherever applicable.

2.1 Summary of Implementation Steps

You should consider these implementation steps when planning your implementation strategy:

1. Implementing Phased Security
2. Planning Emergency Procedures
3. Implementing Default Protection
4. Implementing by Facility
5. Developing Tape Protection
6. Protecting Special Resources
7. Logging and Reporting Options
8. Defining Audit Requirements
9. Developing Security Maintenance Procedures

The remainder of this chapter discusses each step in detail.

2.2 Implementing Phased Security

In a typical security environment, the entire installation will gradually move from DORMANT to FAIL mode. This is called *phased implementation*, and it allows you both to introduce your users to how CA-Top Secret will impact their daily job performance, and how to customize your security database and security environment as the need arises.

There are four modes of implementation available with CA-Top Secret: DORMANT, WARN, IMPLEMENT, and FAIL. These modes can be assigned to the site, to a facility, to a profile, to a user, to a resource, or to an event. Using modes gives you great flexibility in designing your implementation strategy. However, because of this flexibility, you can easily find yourself in the middle of a very complex implementation if you don't take the time to review your options and plan your strategy accordingly.

It is recommended that you initially organize your implementation effort into manageable segments. For example, you could address the implementation by facility because each facility will have different considerations and concerns. This will help ensure that you don't overwhelm yourself, or your security implementation team, with the full implementation effort all at once. The implementation can follow the same strategy that you have used in performing your user and resource inventory.

Most organizations will choose a phased approach to their implementation, especially if the organization must protect a variety of users and resources in different facilities. Even a small organization may choose a phased implementation strategy to address the implementation in manageable segments.

2.2.1 DORMANT Mode

The first mode used in any implementation is DORMANT mode. This is the recommended mode in which to introduce yourself and your organization to CA-Top Secret by inputting your Security File structure through the TSS command. Although CA-Top Secret is installed, it isn't actively validating access. Once you begin to define users or protect resources, and you wish to bring them under CA-Top Secret control either for research or for actual protection, it is time to choose one of these implementation modes: WARN or IMPLEMENT.

2.2.2 WARN Mode

WARN mode provides an excellent tool to determine which users are accessing which resources, or to test the access definitions that you have made in DORMANT mode. As with any other mode, you can set WARN mode for the entire site or for the subset of the organization that you wish to test. Therefore, WARN mode could be set by facility, by profile, by user, by resource, or by event. In WARN mode, CA-Top Secret won't stop violations, but will log those violations, and optionally send violation messages to the user depending on the setting of the MSG control option.

Signon Violations: WARN mode basically emulates FAIL mode in that all users must be defined to CA-Top Secret or signon violations will be generated and recorded. However, WARN mode will not prevent an undefined user from signing on or gaining access to a protected resource.

Password Violations: WARN mode won't prevent a defined user from signing on with an incorrect password, but it will generate a password violation for that user.

Tip: It is recommended that you set the WARNPW suboption of the FACILITY control option. This will force a defined user to supply a correct password in WARN mode.

Note: CA-Top Secret security administrators must always supply a correct password, even in DORMANT mode.

Resource Violations: If you give default protection to specific resource classes by attaching the DEFPROT attribute, WARN mode will generate violations for all resources that have not been defined.

Global WARN Mode: Global WARN mode is most often used to test segments of the implementation, or to back off from FAIL mode when an implemented segment of the organization is in trouble. Your organization could choose an implementation strategy that includes installation-wide use of WARN mode.

2.2.3 IMPLEMENT Mode

IMPLEMENT (IMPL) mode treats defined users in FAIL mode and undefined users in DORMANT mode. This allows you to combine DORMANT and FAIL modes easily in your implementation strategy. In IMPLEMENT mode, all resources which have been defined to CA-Top Secret are protected and can't be accessed by undefined users unless permissions have been defined to the ALL record. Unauthorized access attempts by defined or undefined users will be failed. Resources that aren't defined to CA-Top Secret are generally not protected.

Many organizations use a global IMPLEMENT mode strategy during implementation and override that mode where appropriate by facility, profile, user, resource, or event.

2.2.4 FAIL Mode

CA-Top Secret is in full control of access requests. All users must be defined and resources protected either by being owned or by DEFPROT protection. Unauthorized access requests will be failed. Using a phased approach, your implementation strategy may include a gradual migration by segment to FAIL mode. Once each segment is tested successfully, your implementation goal should be to have your entire installation operating in FAIL mode.

Failing Accesses: You can fail resource accesses through the ACTION parameter on the TSS PERMIT command function, or through the DRC control option in the following ways:

- The ACTION parameter allows you to put a specific permission in FAIL mode no matter which mode the user is in. This allows you to protect critical resources from users who are in the implementation process but aren't yet in FAIL mode, or from undefined users. For more information on the ACTION parameter, refer to Chapter 6, *Resource Security Validation*.
- The DRC control option allows you to specify that selected violations, if incurred, will always fail the attempt. This allows you to fail specific unauthorized access attempts even if the user is in DORMANT or WARN mode. For more information on the DRC control option, refer to the *Control Options Guide*.

2.2.5 Using Concurrent Security Modes

Security modes can be assigned either implicitly—through the MODE control option and suboptions of the FACILITY control option—or explicitly—through the TSS PERMIT command.

When the security mode is set through a control option, that mode applies to either the entire installation or, in the case of the FACILITY/MODE suboption to a specific facility (for example, TSO). In the following example, you can use the MODE control option to place the entire installation in WARN mode.

```
TSS MODIFY MODE(WARN)
```

In the next example, With the MODE suboption of the FACILITY control option, CA-Top Secret treats the BATCH facility as in IMPL mode.

```
TSS MODIFY FAC(BATCH=MODE=IMPL)
```

There are three ways to select and change control options:

- through the CA-Top Secret Parameter File when CA-Top Secret is started
- from a VSE console through the VSE MODIFY TSS command
- through the TSS MODIFY command function.

Each method will be discussed in detail in Chapter 4.

When the TSS PERMIT command function is used, the security administrator can assign a specific security mode to a particular user or profile ACID. In the following example USER01 is put in WARN mode, regardless of what mode the rest of the site is in.

```
TSS PER(USER01) MODE(WARN)
```

Note: ACTION(FAIL) used with the TSS PERMIT command function overrides a user mode as illustrated above.

2.3 Planning Emergency Procedures

It is highly recommended that you plan emergency and troubleshooting procedures before you need them. An emergency isn't the time to try to determine how to approach a problem.

This section discusses some of the more common emergencies you might encounter. It also discusses the troubleshooting tools provided by CA-Top Secret and some recommended troubleshooting procedures.

2.3.1 Production Security Violations

Production abends caused by security violations can occur at any time and when least expected. They can occur if someone has revised a production procedure without requesting the appropriate CA-Top Secret definition revisions, or if an untested change to a CA-Top Secret definition has been made. In any event, the appropriate authority must be available at all times to make the production piece operational.

If production problems occur during working hours, the security administration staff must be prepared to respond to emergency production problems so as to not impact the production schedule. At least one member of the security administration staff should be available at all times to analyze and resolve the problem as quickly as possible.

2.3.2 Emergency ACIDs

If production problems occur off-hours when the security administration staff is unavailable that require the assistance of the systems or applications programmers (who shouldn't normally have the authority required to handle the production problem), procedures can be established for the use of **emergency ACIDs**. An emergency ACID is an ACID that is used to solve off-hours production problems. The operators can assign a emergency ACID to the abending job which will allow the job to process but will record the access activity of the job.

There are two approaches you can use to design emergency ACID procedures.

1. An emergency ACID can be created which has powerful authority to access production resources. Basically, all production resources are permitted to this ACID. When defining a emergency ACID in this manner:
 - Audit this ACID at all times, so that your report will show when it is used and what the emergency ACID is accessing.
 - Do not use any of the security bypass attributes when creating the emergency ACID as an easy way of defining access to all production resources.
 - In case of an emergency, you can use a modified BYPASS(jobname).

Note: The security bypass attributes turn off auditing for the resource class selected for that ACID.

2. An emergency ACID can be created in WARN mode with limited authority to access production resources. Only globally available production resources that aren't defined in the ALL record should be permitted to this ACID. This includes resources such as production load libraries or global production programs.

When using a emergency ACID in this manner only the violations will be recorded. Be sure to review the regular emergency ACID report to monitor the use of this ACID.

Define Multiple Super ACIDs:

You can choose to define a series of emergency ACIDs, possibly by production application, or you can define just one for all production emergencies. The more precisely you can define your emergency ACIDs, the easier it will be to monitor the abnormal, audited, or violation activity. Whichever methodology you use, you should clearly define and explain all appropriate emergency ACIDs to your operations staff so that they are prepared for any production emergency.

Restrict Use to Off-Hours: You should also restrict the use of these ACIDs to off-hours through CA-Top Secret date and time controls. This is one way to discourage misuse of the emergency ACIDs for non-emergency situations.

Monitor Reports: Another way to discourage misuse is to carefully and regularly monitor the emergency ACID reports that you should produce to determine whether the use of the ACID is legitimate. You should also research the cause of the unexpected violation to ascertain that it was not a deliberate attempt by someone outside of the operations area to bypass CA-Top Secret. This is possible if the use of production emergency ACIDs is common knowledge.

The operations staff should be made responsible for the use of these ACIDs and should be held accountable for their misuse.

Note: It isn't a good idea to allow a user who is disgruntled or near termination to use an emergency ACID, since they might maliciously or destructively use the ACID and leave your organization before you can review what they have done.

2.3.3 CA-Top Secret Software Problems

One important characteristic of CA-Top Secret is that as a security system it is, by nature, a critical part of your operating environment. CA-Top Secret is designed to recover from virtually any system error or abend. However, loss of CA-Top Secret's files or major operating system errors can cause system failure. Almost exclusively, errors that cause CA-Top Secret unrecoverable problems will be in the area of file destruction, and for this reason recovery procedures must be an integral part of your security and operating system plan.

Another important element is that CA-Top Secret will make every effort to protect itself from attack, and will, if necessary, disable the operating system before allowing unauthorized security bypasses to occur. If for any reason, known or unknown, CA-Top Secret behaves in an unexpected manner, you should be prepared to solve and circumvent system problems in an emergency situation. The following points may be useful in such a situation:

- CA-Top Secret has secure means of bypassing selected parts of, or the entire, security system. Be familiar with them and test their operation occasionally. Also be aware of the behavior the various facilities will show when bypasses are in effect and be prepared for them. For example, when being bypassed TSO signons revert to the use of the UADS password.
- CA-Top Secret will recover virtually all internal errors. Be sure that procedures exist for printing snap dumps taken by CA-Top Secret and delivering this information to the proper systems areas in a timely fashion. All too often a growing problem, such as a failing DASD controller assigned to the CA-Top Secret files, will go unnoticed because the dumps generated from CA-Top Secret recovery are ignored.
- Most problems with CA-Top Secret will involve security authorizations not working in the manner expected. For these cases, you must have the *Troubleshooting Guide* available and a terminal present so that you can conveniently use CA-Top Secret diagnostic tools when contacting CA-Top Secret customer support. If the preparation for problem diagnosis is proceduralized, this will result in faster, more accurate resolutions from CA when needed.
- If CA-Top Secret has completely failed, a system IPL without CA-Top Secret may be in order. IPL processes to accomplish this should be set up, but should only be known to a select group of trusted employees. However, if there is more than one CPU in your complex, this may be unnecessary if the unaffected CPU can be used to address the problem.
- Since any problem can occur off-hours, contact lists and phone numbers of your security personnel and for CA emergency support should be available. All responsibilities should be clearly understood by all participants.

Troubleshooting Guide:

A number of troubleshooting tools are provided with CA-Top Secret to assist in diagnosing the problem situation. These tools are documented in the *Troubleshooting Guide*. Study these tools so that you will be able to use them quickly to assist CA-Top Secret customer support in an emergency situation.

Before contacting CA-Top Secret customer support, be sure that you are prepared with all proper information about the problem. This will help the customer support staff accurately and quickly diagnose your problem and provide a solution. Review the *Troubleshooting Guide* for details on CA-Top Secret customer support procedures.

2.3.4 Disaster Recovery

Disaster recovery plans are usually fairly involved and complex. It is a huge task to move an entire data center to a new location under emergency conditions. When you begin to implement CA-Top Secret, your disaster recovery plan should be modified to include procedures to bring CA-Top Secret to the disaster recovery site.

If Installing Operating System at Site: If your disaster recovery site permits you to install your own version of the operating system, be sure to plan to bring the CA-Top Secret load library and files to the site. CA-Top Secret should be brought up after IPL, just as it is at your main site.

If Using the Site's Operating System: If the site provides you with an operating system, be sure that you can install CA-Top Secret at that site as part of your disaster recovery operation. Since CA-Top Secret installs quickly, your disaster recovery procedures shouldn't be impacted by the installation of your security software.

Plan to Install Security: No matter which type of disaster recovery site you are using, you should plan to include the installation and use of CA-Top Secret as part of disaster recovery testing. The time of a disaster isn't the time to try to bring CA-Top Secret along for the first time.

Some organizations choose to leave security products out of disaster recovery plans. They feel that in a disaster, they shouldn't have to worry about security. This of course leaves the data center exposed. Also, knowledge of this omission might encourage sabotage against your main data center to force you to relocate to an unsecured operation.

2.4 Implementing Default Protection

You can set up full default protection for specific predefined resources in the Resource Descriptor Table (RDT) Record by attaching the DEFPROT attribute to the particular resources using the TSS REPLACE(RDT) command function. For more information on the TSS REPLACE(RDT) command function, refer to 8.1.2.2, “Changing Values in the RDT” on page 8-7.

For example, the command function shown below tells CA-Top Secret to automatically protect CICS journal control tables (JCT).

```
TSS REP(RDT) RESCLASS(JCT) ATTR(DEFPROT)
```

Keep in mind that many resources don't require full default protection (for example, if you don't want a security violation to occur when you access a particular resource).

Note: You can also dynamically define resources to the Resource Descriptor Table and, in turn, give them default protection. For more details, refer to 8.1.2.2, “Changing Values in the RDT” on page 8-7.

2.4.1 Default Data Set Protection

CA-Top Secret protects data sets and volumes by default in FAIL mode. In the implementation modes, CA-Top Secret allows you to gradually define your data sets so that you aren't required to define all data sets at once. This simplifies your implementation strategy.

To protect all data sets in WARN and IMPLEMENT modes attach the DEFPROT attribute using the TSS REPLACE(RDT) command function. For more details, refer to 8.1.2.2, “Changing Values in the RDT” on page 8-7.

2.4.2 Migrating to FAIL Mode

If you have not added the DEFPROT attribute to protect all data sets in WARN and IMPLEMENT mode, users in WARN and IMPLEMENT modes can access or create data sets that have not been defined to CA-Top Secret.

However, users in FAIL mode can't access data sets that have not been defined to CA-Top Secret because the access and creation of data sets are protected by default in FAIL mode. This suggests the two following migration strategies.

1. If you decide to use WARN mode implementation prior to FAIL mode, it is recommended that you add the DEFPROT attribute just before migrating your installation to FAIL mode. This tells CA-Top Secret that default data set protection is in effect in WARN mode and that users aren't allowed to access data sets which have not been defined. Because WARN mode will only issue violations and won't stop access, this gives you an opportunity for last minute definition of data sets that you may have overlooked during your implementation.

Note: This approach will work only with a global WARN mode implementation.

2. If you have chosen an IMPLEMENT mode implementation strategy, you might wish to gradually migrate your users to FAIL mode to ensure that they aren't accessing undefined data sets. Once you have ascertained that a group of users is properly defined and has access to the appropriate data sets, you can migrate those users to FAIL mode by permitting the mode either to the user or to the group profile.

2.5 Implementing by Facility

Since each facility used in your organization has different implementation considerations, it is recommended that you address them separately. You will find that each facility has its own special resources and a completely different base of users.

Your own environment determines which facility(ies) you will address first. Or, you might choose to address the simpler facilities first so that you can "get your feet wet" with a simple task. You might choose to address the most critical facility first—a facility, for example, that outside users might access.

If your implementation task force can accommodate it, you might wish to address multiple facilities concurrently, with different team members assigned to different facilities. This approach requires good communication and planning so that each facility implementation compliments the others, and so that users affected by more than one facility are getting consistent information and instruction from the different members of the implementation task force.

Since CA-Top Secret allows you to select modes by facility, you can choose whichever approach suits your environment. As you progress, you can migrate the fully implemented facilities to FAIL mode while the other facilities remain in DORMANT or in one of the other implementation modes. Once again, you have tremendous flexibility in planning your implementation, and you should study your options before you proceed.

For detailed information on a specific facility, refer to the *Implementation: CICS Guide*

The facility implementations addressed in this section include:

- BATCH
- CICS
- CA-IDMS
- Other Facilities

2.5.1 BATCH Considerations

The BATCH facility is probably the easiest facility to address, particularly for production jobs, because the resource requirements for these jobs have already been defined. In BATCH, the most common resources protected are data sets, volumes, and programs.

Your implementation strategy can follow the recommendations in the prior discussion of modes. However, combining the IMPLEMENT mode with one of the ACID derivation capabilities may offer the most flexibility in implementing the BATCH facility. CA-Top Secret will derive an ACID for BATCH jobs that are submitted through an online facility, such as CICS or via a // ID CARD in submitted JCL, allowing the batch job to run with the same ACID as the ACID of the online user. This is an effective approach to protecting batch testing for online users. Refer to the *Implementation: BATCH, STC and APPC Guide* for a complete discussion on ACID derivation.

Production Scheduling: CA-Top Secret allows online users to submit batch jobs that will run under an ACID other than the ACID of the online user (called a secondary ACID). The user's ability to submit jobs that will run under secondary ACIDs is the key to effective security for production scheduling. You don't have to allow the production scheduling area to access all production resources at the required access levels. You simply permit the schedulers to submit batch jobs for the ACIDs that you have defined for each production job. If you have specified a password for each ACID, you will have the protection of a password and won't have to divulge the passwords to the schedulers.

This same strategy can be applied to automated scheduling software. You can permit the ACID under which the automated scheduler runs to use the ACIDs that you have defined for your production processing. This allows you to control what the automated scheduler can submit.

Caution Important!

Do not use the attribute NOSUBCHK to give the schedulers (live or automated) the ability to submit production processing. This attribute will allow them to submit under any ACID—including critical systems ACIDs and emergency ACIDs. You will also not be able to audit the schedulers' ACIDs to track the production submitted by each scheduler.

Protecting BATCH Jobs: The major consideration in implementing the BATCH facility is controlling access from batch jobs whether they are input through a card reader or a remote station. The goal should be to define a specific ACID for each individual batch production job or each group of batch production jobs. Although it might sound like an enormous task, it doesn't take much time to code a USER= parameter on the JOB statement for each job in your production JCL library. It is a task that can be handled by any clerk in your organization.

You then have the flexibility of specifying the appropriate security for each job in your site. You can group jobs together under the same ACID without tying yourself into other information on the JOB statement which might not be appropriate. You can specify special ACIDs for critical processing, such as payroll master file updating. You can also easily modify the security required for a specific job.

Moreover, this approach is the simplest and easiest to understand. You don't have to determine which ACID will be derived for a job based on your control option settings. The appropriate ACID will be coded on the JOB statement.

You can choose to use derived ACIDs as a permanent strategy in protecting batch processing, or you can migrate to specific ACIDs coded for each job. Whichever approach you take, the derived ACID or the specifically coded ACID must exist or the batch job may abend.

CA-Top Secret has a number of options available for controlling how a batch job will access the system. These options include:

- Defining ACIDs through the DEFACID suboption

You can define a default ACID for the BATCH facility using the DEFACID suboption of the FACILITY control option. You can then define an ACID, or derive an ACID from the reader/terminal from which the job was submitted. This derived ACID will be applied to any job without a valid ACID. Combining the derived ACID with IMPLEMENT mode allows you to gradually define your batch processing. As you define a valid ACID for each job, the job will no longer pick up the default ACID.

It is strongly recommended that the default be removed after you have completed your BATCH facility implementation to ensure that new batch processing can't be introduced without being evaluated for the appropriate security controls.

- Deriving ACIDs from the JOB statement

To initially minimize required JCL revisions, you can set the JOBACID control option to derive an ACID from information on the existing JOB statement. As with default ACIDs, the use of JOBACID is recommended as an implementation strategy to be replaced with specific ACIDs when the BATCH implementation is complete.

BATCH Passwords:

You can also assign passwords to BATCH job ACIDs. You can choose to have no password on the job, or you can choose to specify a password that won't expire.

Note: It is inconvenient to change a password in the BATCH facility. If you define a password that will expire, you might cause an unexpected interruption in your batch processing. Specifying a password that doesn't expire is a secure method for protecting batch processing, if your batch processing is submitted through an online facility.

2.5.2 CICS, DL/1, and CA-IDMS Considerations

Considerations for implementing CA-Top Secret for CICS, DL/1, and CA-IDMS are similar because the security interfaces for these facilities behave in a similar fashion.

The following table shows the relationship among the resources appropriate to each facility.

Table 2-1. Resources By Facility			
TSO/CA-Roscoe	CICS	DL/1	CA-IDMS
Commands	Transactions	Transactions	Task Codes
Programs	PPTs	PSBs	Sub-Schemas & Programs
Data Sets	FCTs, DCTs, JCTs, TSTs	DBDs	Area

Note: The resources in CICS, DL/1, and CA-IDMS that are similar to data sets are treated as different resource types by CA-Top Secret. The data sets that will be made available to the regions of the different facilities will be opened directly by the regions, and not by the users signing on to those regions. The region ACID will control access to the data sets, and the user ACID will control access to the resource type appropriate to that facility—for example, FCTs within CICS.

Implementation Strategy: CICS, DL/1, and CA-IDMS lend themselves to a gradual implementation strategy, since they usually run in multiple regions in a given organization. Minimally, there are test and production regions to allow for proper testing and debugging of online applications and subsystem software before migration to a production environment. Larger sites may additionally have user acceptance testing regions for end-user testing before migration to production.

It is recommended that you plan your implementation by region, implementing CA-Top Secret in the test regions first and migrating into production. In fact, you can enter your CA-Top Secret definitions through BATCH before you even interface CA-Top Secret with your first selected region.

The same general mode strategies apply for all of these facilities. An effective approach might be to put the region in IMPLEMENT mode so that defined users are controlled through profile access definitions and undefined users are controlled through the security native to the particular facility. As you define new sets of users to CA-Top Secret, you can test each set by assigning WARN mode to its profile. You can revoke WARN mode from the profile when you are satisfied that the users are properly defined. They will then be effectively treated as though they are in FAIL mode. Once all of your users are defined and selected resources protected, you can migrate the region to FAIL mode.

The DEFPROT attribute can control whether CICS, DL/1, and CA-IDMS resources are protected by default or by definition in all modes.

If you choose **not** to give default protection to them, you can include a gradual implementation strategy for resources. An effective approach is to protect users and resources by application. This approach allows the security administration group to concentrate on the requirements of each application, and to design a security scheme that is most effective for that application. Even if the region is already in FAIL mode, you can continue to protect resources by testing the definitions with ACIDs in WARN mode, and then making those resources available to the appropriate users.

CICS, DL/1, and CA-IDMS Native Security: Each of these facilities has its own native security that will coexist with CA-Top Secret in specific modes. In DORMANT mode, each facility has complete control over access through the native security available in that facility. In the implementation modes, WARN and IMPLEMENT, native security and CA-Top Secret security can, in most cases, coexist.

Since CA-Top Secret provides all of the controls available with CICS native security, CA-Top Secret replaces CICS native security in FAIL mode. DL/1 and CA-IDMS provide additional native security features; for this reason, CA-Top Secret will coexist with DL/1 and CA-IDMS native security in FAIL mode, allowing further levels of control. Of course, you can choose to turn off the DL/1 and CA-IDMS native security if this is appropriate for your organization.

See the appropriate *Implementation Guide* for more information on CA-Top Secret coexistence with native security.

Transaction versus Resource Level Security: One of the important considerations in implementing security in CICS, DL/1 or CA-IDMS is transaction versus resource level security. There are advantages and disadvantages to each approach.

Transaction Level Security

Transaction level security is effective for end-users only. Any programmer can design an application to link to the program behind the transaction and get into the protected application by bypassing the transaction level security. Further, if the files available within the facility aren't protected, a programmer can easily modify her program to access the files available behind the transaction level security. Proper program change controls limit these types of exposures. If you choose to implement transaction level security with LCF, consider the following:

1. Since transactions (or task codes for CA-IDMS) are the most obvious element of the application, the resource analysis is easy and straightforward. It is simple to determine which users require which transactions, and to define those requirements to CA-Top Secret.
2. Transactions using LCF aren't ownable resources. Therefore:
 - a. You can't use the TSS WHOHAS command to determine who has access to the transactions.
 - b. Transactions don't fall under the control of administrative scope. As a result, you can't effectively decentralize their administration. Either a CA-Top Secret security administrator at any level can administer all transactions, or he can't administer any transactions.

If you choose to secure transactions through OTRAN, consider the following:

1. Transaction ownership is global. Once a transaction is owned it must be administered across all facilities.
2. OTRANs are general resources and TSS WHOHAS can be used to help administer them.

Resource Level Security

The following considerations apply to resource level security:

1. It is more difficult and time-consuming to determine the resource access requirements for a given application. It is often necessary to involve the applications development group to design effective resource level security.
2. CICS, DL/1, and CA-IDMS resources are ownable resources, on. therefore:
 - a. The TSS WHOOWNS and TSS WHOHAS commands can be used to determine who is responsible for, and who has access to, these resources.
 - b. Administrative scope applies, and administration of resources can be effectively decentralized.
3. Resource level security is the most secure level of security, because it prevents programmers from linking into protected programs or accessing protected files within these facilities.

You may find it appropriate to plan an implementation strategy that combines both forms of security. Many organizations initially address transaction level security, and as time permits gradually implement resource level security. Since these facilities allow gradual implementation of resources, even in FAIL mode, this may be the most effective approach for the implementation of CICS, DL/1, and CA-IDMS security for your installation.

2.5.3 Other Facilities Considerations

When implementing other facilities, you will find that the considerations are similar to the considerations for the facilities discussed previously. It depends on the nature of the facility and how it goes about providing security for users and resources defined to it.

Review the *Implementation: Other Interfaces Guide* for more information on the implementation of additional facilities.

2.5.4 APPC/VSE Considerations

There is one basic step for using CA-Top Secret to secure APPC conversations:

Specify which LUs can establish sessions by defining remote and partner LUs to the CA-Top Secret APPCLU record.

For more information on APPC/VSE, refer to the [Implementation: Batch and APPC Guide](#).

2.6 Developing Tape Protection

Tape protection can almost be considered a separate implementation. You can plan to protect tapes by volume or by data set name. For a more detailed explanation of tape protection, refer to Chapter 7 under the section called: *Protecting Tape Volumes*.

CA-Top Secret allows you to define volume level security for tapes. This strategy is effective only in small sites where the creator of the tape file can be considered the owner of the tape volume. When the tape is scratched, ownership must be removed so that the volume can be assigned to a new user.

There are several options available to provide data set name level security for tapes. These options are:

- Tape Data Set Security
- External Tape Management Packages
- CA-Top Secret Tape Management Interfaces
- Bypass Label Processing

Tape Data Set Security:

CA-Top Secret can provide limited tape data set name security through the TAPE(DSN) control option. The protection is limited due to limitations within VSE. VSE records only the last 17 characters of the data set name on the tape label, and verifies that these last 17 characters match the last 17 characters of the data set name coded in the JCL. VSE calls CA-Top Secret using the full 44 character data set name coded in the JCL and allows access based on the results of this security call. As long as the data set names in your environment don't exceed 17 characters, TAPE(DSN) can provide effective tape data set control. For tape data sets that have data set names longer than 17 characters, it is possible to bypass TAPE(DSN) security by prefixing the data set name in the JCL with an authorized prefix while the actual data set name isn't authorized to the user.

External Tape Management Packages: If you are using a tape management package such as CA-Dynam that provides integrity for full 44-character data set names, then any of these packages with TAPE(DSN) can provide effective security for tape data sets. However, this strategy won't protect the label bypass mechanisms available with the tape management packages. .cp

CA-Top Secret Tape Management Interfaces: The most secure method of tape security is to use one of the existing CA-Top Secret tape management package **interfaces** for CA-Dynam. This allows you to control tape data sets by the full 44-character data set name, and to control the label bypass mechanisms available with these tape management packages.

| **Bypass Label Processing in OS/390:** One last consideration in tape protection is
| Bypass Label Processing (BLP) for OS/390. CA-Top Secret restricts the use of BLP
| unless it is specifically authorized to a user. It is recommended that BLP only be per-
| mitted where absolutely necessary to limit this exposure.

2.7 Protecting Special Resources

Certain resources, such as CPUs and terminals, require special considerations because of the nature of the impact they may have on your environment. For this reason, it is recommended that you have most of your implementation in place before you address these resources.

2.7.1 VSE Libraries

CA-Top Secret includes the capability to secure VSE libraries at the member level. Using this feature, the reading or updating of selected members can be secured or audited. Member level protection in CA-Top Secret compliments rather than replaces library security. Regardless of the member level protection, to read a member, a user must first have READ authority for the library. Likewise, regardless of the member level protection, to update a member, a user must first have UPDATE authority for the member. For details, refer to the *Implementation: Batch and APPC Guide*.

2.7.2 CPUs

CPU control can be very effective for restricting access to CPUs in your site. Once you have defined a CPU or all CPUs by prefix, no user will be able to sign on unless they are specifically permitted to access the CPU. CPU control should be carefully designed and planned before implementation to ensure that you don't unwittingly lock your users out of accessing the systems they need to do their job. .cp

2.7.3 Terminals

CA-Top Secret allows you to protect terminals so that they can only be accessed by authorized individuals. While this is often a popular implementation strategy, global terminal control often poses administrative problems that surpass the value that you might receive from terminal control.

CA-Top Secret Protects Ports, Not Hardware: When protecting terminals, you aren't protecting the physical hardware device. You are protecting the port through which the terminal is defined to your system. It doesn't matter which actual physical device is attached to that port. The name of the port that you protect is defined and can be changed by your network control area.

This can be the cause of administrative problems. If you choose to protect all, or a significant number, of terminals in your organization, you must have solid procedures between the network control area and the security administration area to ensure that the security administration area is notified of every planned change to the network. This gives the security administration area time to redefine the affected terminals to complement the network change so that appropriate security is in place when the network is reconfigured.

Limit Protection to Critical Terminals: It is recommended that you limit the number of terminals protected to those that are truly critical, such as dial-in ports, production scheduling ports, or end user ports for sensitive applications. This should make your terminal maintenance more manageable.

Remember, CA-Top Secret architecture is based on the user, and it is the user who is responsible and accountable for her actions no matter which terminal she uses to access your corporate resources.

Source Control: Source control allows you to restrict users to specific sources of entry, such as terminals or readers. You will probably not be required to restrict most of your users to specific sources of entry. However, there will be a critical subset of your users for which this control may be appropriate, such as production schedulers or end-users using sensitive applications.

The same considerations given to terminal protection should be given to source control because you will be defining the port as the source of entry. Again, it is critical to coordinate network reconfigurations between the network control area and the security administration area, or an unsuspecting user may come in to work one day only to find that he can no longer sign on to his terminal.

2.7.4 Record Level Protection (RLP)

CA-Top Secret lets you extend file level security by allowing you to protect records within a file and even fields within a record. This capability gives you detailed control over which users have access to the data within your system.

The information used to protect records and fields is stored in a reserved ACID called the Static Data Table (SDT). The SDT is made up of record elements that you define to protect certain records and fields. For more information on the SDT and how to create and maintain its record elements, refer to Section 8.3, the Static Data Table (SDT) Record.

Before you define the records and/or fields to be protected, there is some preliminary information you must gather. This information and the complete procedure for implementing Record Level Protection is covered in Section 7.6, Protecting Records and Fields.

2.7.5 Screen Level Protection (SLP)

In addition to protecting data at the record and field level, you can also protect data from being displayed or updated on a user's screen. This type of protection is called Screen Level Protection (SLP) and is used to protect fields within a CICS map.

Note: SLP takes effect before the application gets control.

The information used to protect data on screens is stored in a reserved ACID called the Static Data Table (SDT). The SDT is made up of record elements that you define to protect certain screen fields. For more information on the SDT and how to create and maintain its record elements, refer to Section 8.3, the Static Data Table (SDT) Record.

Before you can define the data on the screen to be protected, there is some preliminary information you must gather. This information and the complete procedure for implementing Screen Level Protection is covered in Section 7.10.4, Implementing Screen Level Protection.

2.8 Logging and Reporting Options

One of the main functions of a security product is to control unauthorized access attempts by users in your data processing environment. An equally important function is to track security violations and other selected activity.

CA-Top Secret provides many options for tracking violations and activity. You should decide, as part of your implementation strategy, which options are best suited to your environment and plan to implement those options.

2.8.1 Logging Activity and Violations

CA-Top Secret provides you with media with which to record security violations and other activity. You can record this information to the CA-Top Secret Audit/Tracking File.

Audit/Tracking File: There are certain advantages to using the Audit/Tracking File instead of SMF. Refer to Chapter 3, “Violations, Logging, Reporting, and Auditing” on page 3-1 for the specific advantages.

Select Type of Activity for Logging: You must select the activity you wish to log. This can be done globally or by facility. Violation activity will always be logged as long as you have included the Audit/Tracking File in the CA-Top Secret startup procedure.

2.8.2 Reporting Activity and Violations

One of the easier ways to monitor violations, or any selected activity, is to develop and produce reports on a regular basis. TSSUTIL, the CA-Top Secret report generator, allows you to produce violation and/or activity reports based on customized selection criteria. The *Report and Tracking Guide* discusses the use of TSSUTIL.

Any other report generator or program can be used to produce customized reports based on violations or activity. The format logged is an AUDIT type-80 format.

Generating Reports: When generating reports, it is a good idea to segment the information you are monitoring, rather than to produce one large report for your organization. If you segment the critical selection criteria from the non-critical, you can more easily focus on critical information. Consider the following sample breakdown of daily reports:

- Report A contains violation information for systems resources.
- Report B contains violation information for payroll and accounts payable resources.
- Report C contains violation information for all other resources.
- Report D contains initiation information for dial-up terminals.
- Report E contains audited activity for critical production files.
- Report G contains audited activity for all security administrators.
- Report H contains violation or audited information for super ACIDs.

While the needs of each organization differ, you can see that when reporting is segmented, as shown in the sample breakdown, it is much easier for the security administrator to review critical violations and activity because the critical information is structured in such a way that it stands out from the less critical information.

Ad Hoc Reporting: You can also produce ad hoc reports that address special situations. For example, if you suspect that one of your users has suspicious access patterns, you might audit the user and produce a one-time report of his activity.

2.8.3 Online Tracking of Activity and Violations

There are two options for online monitoring of violations and activity:

- Operator consoles defined for route code 9 messages, and
- TSSTRACK, the CA-Top Secret online monitoring utility.

Route Code 9 Consoles: CA-Top Secret allows you to specify that CA-Top Secret violation and initiation messages be sent to any operator console defined for route code 9. This capability allows the operations staff to monitor violations—which can be important, especially after working hours when activity isn't being monitored by security administrators. The operators always have the option of suspending users who appear to be generating excessive violations, or at least of notifying the appropriate authorities.

If you wish to use a remote route code 9 console as your security console for use by the security administration staff, it is recommended that you define the console so that commands can't be entered from the remote location. This avoids the exposure to the machine room operations area from remotely entered console commands.

Note: This is done through the LOG options.

TSSTRACK: Some organizations may choose to use a route code 9 console as "the" security console for online monitoring of violations. TSSTRACK, however, is a more powerful and flexible tool for providing the security console function at any TSO or CICS terminal, if the terminal is used by an authorized CA-Top Secret security administrator.

TSSTRACK reviews the information on the Audit/Tracking File. Therefore, the Audit/Tracking File must be part of the CA-Top Secret startup procedure if you wish to use TSSTRACK as your online monitoring tool.

See the *Report and Tracking Guide* for details on the use of TSSTRACK.

2.8.4 User Message and Violation Suppression

CA-Top Secret provides several options for suppressing messages and violations.

- You can suppress sending messages to users in WARN mode by not specifying the MSG suboption of the LOG control options.
- You can suppress sending selected messages to users through the MSG control option.
- You can suppress selected violations issued to users, but not the logging of violations, by changing characteristics of detailed violation reason codes through the DRC control option.

Suppressing messages may be an acceptable implementation strategy, particularly during the testing stages. It is recommended, however, that you avoid message or violation suppression if possible. User messages and violations are often an important debugging tool when you are trying to resolve user problems. If you find that you must suppress mes-

sages, you should cautiously choose the messages or violations that you are going to suppress and communicate to any person who handles security problems that users may not be receiving the selected messages or violations.

For information on the no logging function available in the CA-Top Secret application interface, refer to Chapter 11 under the title: 11.1.2.1, “General Resource Checking” on page 11-9.

2.9 Defining Audit Requirements

Your internal auditors should monitor both the effectiveness of the security implementation and adherence to any policies that have been defined. In most organizations, the EDP auditors perform this function. Their responsibilities toward the security effort should be clearly defined in your security policy.

CA-Top Secret has been designed with a number of capabilities that allows auditing the implementation and effectiveness of CA-Top Secret. This chapter explains how best to put these capabilities to work. .cp

2.9.1 Defining CA-Top Secret Auditors

The auditors should be defined to CA-Top Secret early in the implementation so that they can work with, and monitor the activity of, the security administration staff. If the security administration staff and the auditors work together throughout the implementation, each strategy can be evaluated before it is implemented. This practice can save implementation time and potential rework.

Defined as Administrators: CA-Top Secret auditors are defined to CA-Top Secret as security administrators and can be defined at any level. You will probably choose to define a central level auditor (SCA) as a permanent account to perform routine security audits. Divisional (VCA) and departmental (DCA) auditors are often temporary accounts defined to allow periodic audits of corporate functional areas as shown in the next figure. For more information on defining auditors as security administrators, refer to 4.1, “Creating Security Administrators” on page 4-2.

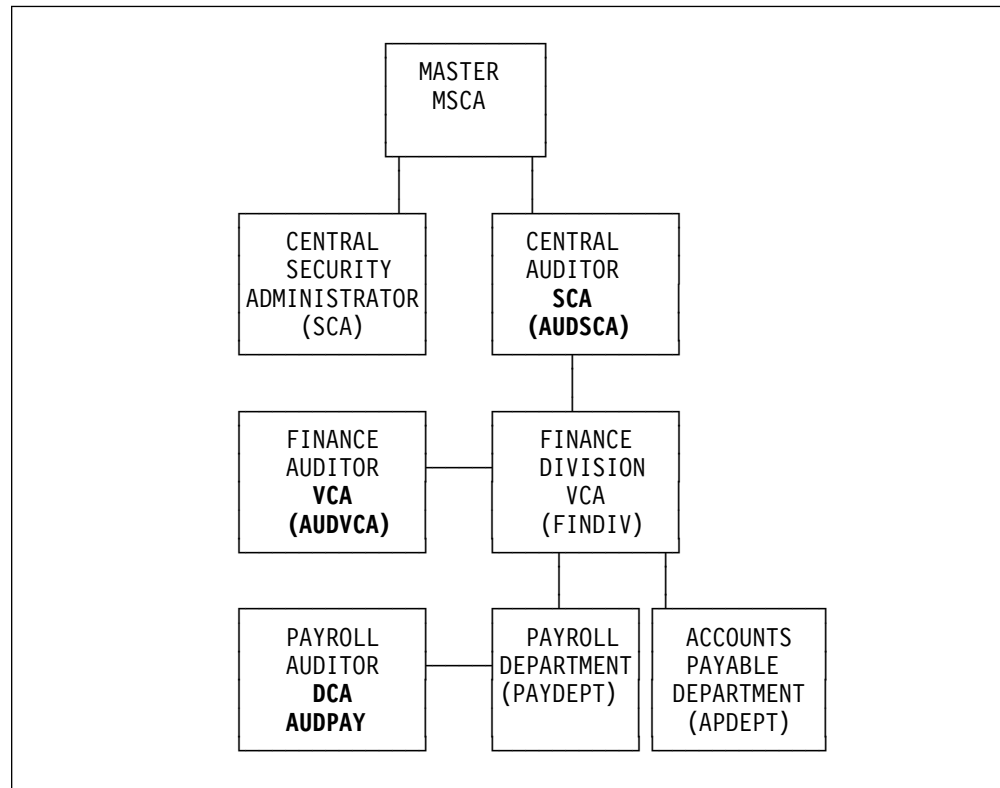


Figure 2-1. Defining Auditors

As with any CA-Top Secret administrator, you must assign the proper auditing authorities to the audit accounts. Typically, auditors are allowed to list information from the CA-Top Secret Security File and use the CA-Top Secret auditing tools.

2.9.2 Using CA-Top Secret Auditing Capabilities

The CA-Top Secret auditing tools are available to any administrator defined with auditing authorities. The security administration staff might use these tools to monitor and to help control their own activity.

The following tools are available to review the CA-Top Secret Security File:

- The TSS LIST, WHOOWNS, and WHOHAS commands allow the auditor to review the Security File for information on specific ACIDs or resources.
- The TSSCHART utility allows the auditor to review file design to ascertain that it has been done effectively.
- The TSSCFE utility allows auditors to create customized reports on information in the Security File using their own report generators.

- The journal file generated by the Command Propagation Facility (CPF) provides the auditor with the ability to monitor TSS commands going to and coming from remote locations.

TSSUTIL and TSSTRACK: The TSSUTIL and TSSTRACK utilities can be used to monitor violations and other activity, including audited activity.

TSSAUDIT: TSSAUDIT is a utility designed to assist the auditor in performing a CA-Top Secret and VSE audit.

These utilities are discussed in detail in the *Report and Tracking Guide* and in Chapter 3, *Violations, Logging, Reporting, and Auditing*.

2.10 Developing Security Maintenance Procedures

Ongoing maintenance should be an important concern during and after the security implementation. This maintenance will take the form of updates to the CA-Top Secret Security File, as well as maintenance to the CA-Top Secret software itself. It is important that your maintenance procedures be in place, so that the approach is defined before receiving the first request.

If you have chosen a gradual approach to security implementation (implementing functional areas and facilities one at a time), maintenance will become a requirement before the implementation is completed. Your maintenance procedures should be designed to anticipate these requirements.

As your environment changes, as is common in most installations, you will be required to revise your security definitions to reflect these changes. It is important to determine that the changes to security definitions are both necessary and legitimate. For this reason, you should have a CA-Top Secret Security File maintenance procedure that allows you to ensure the requested revisions are correct and authorized.

2.10.1 Verifying Change Requests

If the organization is small, and the security administration staff can easily identify and control all users and resources, then the central administrators might be able to verify the requests for changes. However, if the organization is large, it is difficult for the central staff to know all users and resources. They will have to depend on other individuals to verify change requests. In large organizations, or even in small ones, it is recommended that the representatives of the functional area that owns the resource(s) be responsible for verifying the necessity and accuracy of change requests. The request should be made in writing with the proper authorization.

Maintenance Request Forms: Many installations design security maintenance request forms that are completed by the appropriate functional area, and are approved by the appropriate functional authority. The forms are then submitted to the appropriate administrator for revision of the Security File. The forms are then filed as a permanent record of the request. These forms should contain all of the information necessary for the revision, including effective date, resource name and level of access required, user or profile name, and expiration date (if the request is for temporary access).

Proper Maintenance: Be sure that your maintenance activity follows your original Security File design. Be careful that your profile structure isn't compromised by numerous requests for updates to user ACID records. Review each request to ensure that the request falls in the appropriate place in the Security File. It is possible that the requestor is unfamiliar with the structure and has requested an update for an inappropriate ACID. You might have to review the request with the requestor and modify it before the update is actually made to the Security File.

Design Procedure for Quick Turnaround: Your CA-Top Secret Security File maintenance procedure should be designed for quick response. The procedure should be designed in such a way that the requestor can receive a quick turnaround for the request. If quick response isn't practical, then the turnaround time for requests should be communicated and understood by all user areas so that they can effectively plan for timely Security File revisions. Of course, emergency procedures should be available for immediate response when required.

The ability of a central security administration staff to respond quickly to maintenance requests can determine whether or not you choose to decentralize CA-Top Secret security maintenance. If certain areas require more timely response than is possible at the central level, you may choose to decentralize maintenance for those areas. Of course, if you have designed your Security File so that the problem areas are already divisions or departments, decentralization is simple. For a detailed discussion of centralized versus decentralized security, refer to the *Planning Guide*.

2.10.2 CA-Top Secret Software Maintenance

CA-Top Secret software maintenance is usually delivered several times a year. You will probably receive your first maintenance tape before your implementation is completed. Therefore, you should be prepared to apply this maintenance during the early stages of the implementation process. This will ensure that your site can run CA-Top Secret software up to its current level of maintenance.

Reasons for Maintenance: Usually CA-Top Secret maintenance is required to introduce new or enhanced security product features into the environment. In this case, an organization can take their time in installing the maintenance.

On occasion, however, CA-Top Secret maintenance is provided to fix system problems. This type of maintenance might be delivered on the quarterly maintenance tape, or you might receive this maintenance from CA-Top Secret customer support in response to a specific problem that your organization is experiencing. An emergency security product maintenance procedure should be available to respond to this type of situation, particularly if your organization is being impacted by these system problems.

Maintenance Considerations: A number of considerations should be taken into account when doing CA-Top Secret maintenance:

1. The maintenance should be scheduled at a time when it will have the least impact on your environment.
2. Test plans should be available to ensure that CA-Top Secret is still functioning as designed for your environment.
3. **Do not** update or zap the CA-Top Secret modules with your own in-house modifications. This negates the integrity of the CA-Top Secret software. In addition, CA-Top Secret contains anti-tampering code to detect unauthorized modifications to CA-Top Secret software and CA-Top Secret will disable the operating system or force re-initialization of an unmodified CA-Top Secret system if such a modification is detected.

2.10.3 System Software Maintenance

In addition to security product maintenance, you should develop maintenance procedures for other systems products that may impact the security system.

VSE Security Interface: CA-Top Secret works through the VSE Standard Security Interface and is rarely impacted by VSE maintenance. Occasionally, IBM maintenance is applied to these interfaces and a change to CA-Top Secret may be required. If you receive early releases or special releases of VSE maintenance that revise these interfaces, you should take special care in applying and testing this maintenance with CA-Top Secret. Testing procedures for operating system software changes and upgrades should always include a verification of basic security system functions as part of the plan.

You should also ensure that maintenance to interfaces of other vendor products still function properly with CA-Top Secret. Testing CA-Top Secret should always be a part of your test plans for supported vendor software.

Chapter 3. Violations, Logging, Reporting, and Auditing

By default, CA-Top Secret logs all violations to the Audit/Tracking File. Here are some advantages to using the Audit/Tracking File:

- All activity that would normally occur can be logged to the Audit/Tracking File.
- Logging to the Audit/Tracking File can't be suppressed. This eliminates a potential security exposure.
- Reports can be generated for up-to-the-minute information.
- You can only use the online tracking capability, TSSTRACK, if you are logging to the Audit/Tracking File.
- TSSUTIL can be used to routinely archive audit tracking information.

If these advantages apply to your organization, it is recommended that you log violations and activity to the Audit/Tracking File.

Security events that are logged can be selectively extracted to produce reports by using the batch utility, TSSUTIL. CA-Top Secret also allows the online, real time display of selected security activity on CICS terminals with the TSSTRACK utility. A powerful tool for auditing, TSSAUDIT, permits auditors to monitor changes to the Security File and sensitive VSE facilities and data areas.

3.1 Violations and Logging

After preventing unauthorized access, the most important function of a security system is to report on who the violators are. The LOG control option is the primary mechanism that controls the reporting of system activity and violations for all facilities. You can also use the LOG suboption of the FACILITY control option to control individual facilities. Among the options that can be selected to track security breaches are:

- A violation will generate a descriptive message sent to the security console and/or the user's online terminal.
- Job/session initiations and terminations, resource accesses, security violations, or any combination of these events will be logged for all or selected facilities.
- A record of selected types of events will be logged to SMF and/or the Audit/Tracking File. The Audit/Tracking File can be shared across CPUs, providing a single reference source for security events.

Note: When ACTION(AUDIT) is used with the PERMIT command function, it will audit all accesses to the specified resource regardless of the mode or logging options of the user.

The logging of activity is completely transparent to the user. In the following example violation messages are logged to SMF or the Audit/Tracking File, and the user isn't notified of violations.

```
TSS MODIFY LOG(MSG)
```

Note: In FAIL mode, messages are issued regardless of the LOG control option specifications.

You can set limits as to how many violations users can accumulate per session, and define an automatic action through the VTHRESH control option. The following example suspends a user upon his seventh violation.

```
TSS MODIFY(VTHRESH(6,SUS))
```

The cancellation and suspension is automatic only for BATCH. For other online users, CA-Top Secret prevents further access of any kind by locking out the terminal. This forces the user to sign off.

Note: Access due to bypassing is always logged.

The following is a brief synopsis of control options that affect the CA-Top Secret logging and message generation routines:

Option	Description
DRC	Allows you to tailor the behavior of violations by Detailed Violation Reason Code. This option, by design, can't suppress the logging of violations to SMF and/or to the Audit/Tracking File.
LOG	Allows you to specify the selected activity that you wish to log.
MSG	Allows you to modify message characteristics for your installation.
VTHRESH	Allows you to set a violation threshold for non-password related violations which, when exceeded, will take a selected action against the user.

These control options and all of their suboptions are detailed in the *Control Options Guide*.

3.2 Logging Security Events

CA-Top Secret provides several batch utility programs designed to help monitor and control system security, log system activity, and perform disaster recovery. Since CA-Top Secret Security, Audit/Tracking, and Recovery Files may be shared between multiple CPUs and operating systems, these batch programs are totally compatible between VSE, MVS, and VM systems.

Detailed information on logging options can be found in the *Report and Tracking Guide*. The next sections describe the following utility programs:

- TSSAUDIT
- TSSCPR
- TSSRECVR

3.2.1 TSSAUDIT

TSSAUDIT is a batch utility that allows Security Auditors (discussed later in this chapter) to monitor changes made to the Security File and sensitive facilities and data areas. More specifically, it can be used to list the following kinds of information:

- Those ACIDs that possess administrative or special privileges (such as AUDIT, CONSOLE) or any of the security bypass attributes.
- The changes made to the Security File. TSSAUDIT generates this information for a given date or time span by examining the Recovery File. All changes made by a particular ACID can also be requested. The ACID must fall within the scope of the administrator running TSSAUDIT.

3.2.2 TSSCPR

The TSSCPR utility is run against the CPF Recovery File to produce a flat file record. This can then be filtered either through the TSSREPORT3 EARL report option or through another report writer to depict the contents of the CPF Recovery File.

3.2.3 TSSRECVR

The TSSRECVR utility is used to aid recovery from loss or corruption of the CA-Top Secret Security File. During normal system operation, CA-Top Secret maintains a record of all changes to the Security File in the Recovery File. The Recovery File is a perpetual file, meaning that changes are recorded to the file in a wraparound format. Therefore, this file must be large enough to accommodate all changes that occur between Security File backups.

3.2.4 TSSRPTST

3.2.5 TSSTRACK

TSSTRACK can be used to monitor security-related events from an online terminal in a real-time manner. This allows the security administrator to monitor suspicious activity "as it happens". Furthermore, TSSTRACK enables all CPUs on a single security audit file to be monitored from a single terminal. In addition to the capability to monitor events in real-time, TSSTRACK can go back to a specified date to focus on a selected facility or CPU, or to focus on violations only.

The events that a security administrator is able to monitor using TSSTRACK are limited by his administrative scope. All the information that TSSTRACK displays is obtained from the CA-Top Secret Audit/Tracking File, which means that only information logged to this file can be monitored.

TSSTRACK can be used from CICS only.

3.3 Running Security Reports

In addition to logging information on resource access attempts and security events, CA-Top Secret provides additional batch utilities designed to translate this activity into various types of reports—from block charts to fixed format output, to pre-formatted easy-to-understand reports (using CA-Earl). A full description of these options, along with sample reports, can be found in your *Report and Tracking Guide*. For now, let's just take a brief look at the following options:

- TSSCHART
- TSSUTIL
- TSSCFEILE
- TSSREPORT
- TSSREPORT2

3.3.1 TSSCHART

TSSCHART is a batch utility which provides an overview of your security database architecture by generating block charts of ACIDs/owned-resource relationships. This allows you to examine your security hierarchy "at-a-glance" and can be very helpful during your initial CA-Top Secret implementation process. Scope restrictions are honored, so that if a properly authorized DCA uses the TSSCHART utility, the block chart he receives will only illustrate the ACIDs and resource relationships within his department. To view the installation as a whole, the utility would have to be run by the MSCA or a properly authorized SCA.

3.3.2 TSSUTIL

TSSUTIL is a flexible report generator/extract utility that is used to provide batch reports of any security-related events logged to the Audit/Tracking File and/or SMF. You can produce precisely focused reports to monitor all kinds of security events. Selection criteria for TSSUTIL include:

- ACIDs
- jobs
- specific resources
- resource types
- facilities
- zones
- divisions
- departments
- dates
- types of access
- CPUs
- violations
- audited incidents

3.3.3 TSSCFILE

TSSCFILE is a batch utility which produces a fixed format output file whose records parallel the TSS LIST command output. This file can then be used by a site or vendor extract tool to write customized reports based on the configuration and content of your security database. As with the other batch reporting options, scope and administrative authority limitations are honored.

3.3.4 TSSREPORT and TSSREPORT2

Under CA-Top Secret VSE, TSSREPORT and TSSREPORT2 use the output of TSSCFILE and TSSUTIL, respectively, to produce pre-formatted CA-Earl reports. There are several different report layouts to choose from and, depending on your current needs and familiarity with CA-Earl, you can design additional layouts.

For more information about these options and sample reports using each of the different pre-formatted layouts, refer to your *Report and Tracking Guide*.

3.4 Auditing

In addition to reporting and tracking facilities, the TSSAUDIT utility allows auditors to monitor changes to the Security File, sensitive VSE facilities, and data areas.

CA-Top Secret doesn't formally distinguish between an auditor and an ordinary CA-Top Secret security administrator (there is no type of control ACID specifically designated auditor). Who is an auditor is strictly a matter of the functional responsibilities assigned to an employee. These functional responsibilities will, of course, be reflected in the types of CA-Top Secret administrative authorities associated with his ACID. The TSS ADMIN command function is used to assign administrative authorities, including audit-type authorities as explained in 4.1, "Creating Security Administrators" on page 4-2. The following example shows how to give an ACID the authority to audit certain resources and certain ACIDs, and to use TSSUTIL, TSSAUDIT, and TSSTRACK to track both types of information.

```
TSS ADMIN(auditor's acid) RESOURCE(AUDIT,REPORT)
                        ACID(AUDIT,REPORT)
```

The ACIDs and resources must be included within the auditor's scope of authority, which is determined by the ACID type assigned him when he was defined to CA-Top Secret. The scope can range from the entire installation (SCA) to just himself (USER).

3.4.1 Auditing Users and Resources

A CA-Top Secret auditor has the authority to audit users and resources.

- To audit users, the auditor attaches the AUDIT attribute to the user's ACID.
- To audit resources, the auditor updates the AUDIT record with the resource or resource prefix he wishes to audit.
- To audit a specific permission, the auditor includes the ACTION(AUDIT) keyword on the PERMIT command function.

The auditor can audit critical resources on a permanent basis and produce reports or monitor online the results of the audit. He can spot-check user activity by periodically auditing key personnel.

The auditor may wish to coordinate his audit activity with the security administrator. The security administrator will be doing concurrent audits to monitor effectiveness of the security implementation.

Note: Carefully select audit criteria and revise this criteria as audit requirements change. This will avoid the unnecessary generation of large numbers of audit records.

For more information on auditing CA-Top Secret and VSE, see the *Auditor's Guide*

Chapter 4. Setting Up and Modifying Your Security Environment

The following topics discussed in this chapter will help you set up and modify your security environment:

- Creating security administrators
- Identifying users by defining ACIDs
- Creating profiles, and fields attached to ACIDs
- Creating groups
- Displaying information
- Changing your security environment using control options

4.1 Creating Security Administrators

As previously discussed in Chapter 1, *Introduction to CA-Top Secret*, there are **six** types of security administrators that are identified by **Control ACIDs**. These types are:

Control ACID	Description
MSCA	Master Security Control ACID
SCA	Central Security Control ACID
LSCA	Limited Central Security Control ACID
ZCA	Zone Control ACID
VCA	Divisional Control ACID
DCA	Departmental Control ACID

The following section explains how to create each type of Control ACID and how to assign administrative authorities to each type.

4.1.1 Adding Control ACIDs

The following parameters need to be specified when defining any new Control ACID to CA-Top Secret:

NAME	Identifies its name. Names can be up to 32 characters in length, must be surrounded by single quotes if embedded with blanks, and can use letters, numbers, and special characters.
PASSWORD	Specifies an up to eight-character password.
TYPE	Identifies the ACID as a DCA, VCA, ZCA, LSCA, or SCA.
SCOPE	Determines the scope of the LSCA.

The following example shows how to define a VCA (DEVVCA) for the Development Division.

```
TSS CREATE(DEVVCA) TYPE(VCA) NAME('DEVELOPMENT VCA')
      DIV(DEV) PASS(VEFOK,7)
```

Only an MSCA can create and define the levels of administrative authority for SCAs and LSCAs. Typically, the MSCA creates an SCA merely to serve as a backup. In other cases, it would be advantageous to create SCAs to function as legitimate corporate-level Security Administrators. The scope of an LSCA is defined through the ADMIN SCOPE keyword (see 4.1.2.10, “SCOPE Authority” on page 4-18). How you configure the scope of authority and levels of administrative authority for your Control ACIDs depends on whether you want to administer centralized or decentralized security administration for your site. For more information on centralized and decentralized security, refer to the *Planning Guide*.

4.1.1.1 How to Create the MSCA

The MSCA is a unique Central Security Administrator:

- There is only one MSCA, while there can be many SCAs, LSCAs, ZCAs, VCAs, and DCAs.
- The MSCA's ACID defined at installation time. All other ACIDs must be created. Only the MSCA can create SCAs and LSCAs.
- Like other Central Security Administrators, an MSCA has unlimited scope; however, only an MSCA has implicit unlimited administrative authority.
- The MSCA's ACID can't be deleted, although it can be renamed.
- The MSCA can log on or initiate with only password checking in force; no expiration, facility, source, or terminal checking is performed by CA-Top Secret.

These types of security checking, in addition to password checking, can be performed on all other Control ACIDs.

Note: All Control ACIDs, including the MSCA, go through CA-Top Secret password checking, even in DORMANT MODE.

4.1.1.2 How to Create an SCA

An SCA is not associated with any specific division or department but, rather, has unlimited scope. Most sites define only a few SCAs. It is recommended that, at a minimum, one "secondary" Central Security Administrator is created as a backup to the MSCA.

In the following example, the MSCA creates an SCA whose password expires the first time he logs on and uses the site's default expiration interval for all subsequent passwords. For more information on passwords, refer to Chapter 5, *Passwords and System Entry Security*.

```
TSS CREATE(SCA1) NAME('Nat Abe1s') TYPE(SCA)
      PASS(TRUST,,EXP)
```

To give a secondary SCA most of the authority of the MSCA enter:

```
TSS ADMIN(SCA1) RESOURCE(ALL) ACCESS(ALL)
      ACID(ALL) FACILITY(ALL) DATA(ALL,PWVIEW,PROFILE,SESSKEY)
      MISC1(ALL) MISC2(ALL) MISC8(ALL) MISC9(ALL)
```

An SCA with this authority can do almost everything except define another SCA or LSCA or change the administrative authority of an existing SCA or LSCA.

4.1.1.3 How to Create an LSCA

An LSCA is, essentially, an SCA whose scope of authority has been limited. That authority can include several zones as well as other LSCAs. Only the MSCA can create an LSCA and determine its scope through the ADMIN SCOPE keyword. For more details on the ADMIN SCOPE keyword, see 4.1.2.10, "SCOPE Authority" on page 4-18.

In the following example, the MSCA creates an LSCA whose password expires in seven days. For more information on passwords, refer to Chapter 5, *Passwords and System Entry Security*.

```
TSS CREATE(LSCA1) NAME('Bill Bailey') TYPE(LSCA)
      PASS(HONOR,7)
TSS ADMIN(LSCA1) SCOPE(ZONE1,ZONE2,ZONE3)
```

LSCAs are an optional level of security control and need not be defined.

4.1.1.4 How to Create a ZCA

Zone administrators can only be established by a Central Security Administrator. Each ZCA is associated with a particular zone. The responsibilities that can be performed by a ZCA include administrative tasks for the divisions, departments, users, and profiles linked under the zone.

In the following example, an SCA creates a ZCA whose password expires when he first logs on; however, by specifying 30 for the second suboption instead of allowing it to default, CA-Top Secret prompts the user for a new password every 30 days. For more information on passwords, refer to Chapter 5, *Passwords and System Entry Security*. For more information on creating zones, refer to 4.2.1.1, “How to Create a Zone” on page 4-20.

```
TSS CREATE(ZCA1) NAME('Bob Baker') TYPE(ZCA)
      PASS(HONOR,30,EXP) ZONE(ZONA)
```

A zone can have several ZCAs or, under a centralized administrative system, no ZCAs. Zones, and therefore ZCAs, are optional levels of security and administration and need not be defined.

4.1.1.5 How to Create a VCA

Divisional administrators can be established only by a Central Security Administrator or a ZCA. Each VCA is associated with a particular division. The responsibilities that can be performed by a VCA include administrative tasks for the departments, users, and profiles linked to this division.

In the following example, a Central Security Administrator creates a VCA whose password does not expire. For more information on passwords, refer to Chapter 5, *Passwords and System Entry Security*.

```
TSS CREATE(VCA1) NAME('Charles Clark') TYPE(VCA)
      PASS(BOLD,0) DIVISION(FINDIV)
```

4.1 Creating Security Administrators

A division can have several VCAs, or, under a centralized administrative system, no VCAs. In the latter case, the administrative requirements for this division would have to be performed by a Central Security Administrator.

4.1.1.6 How to Create a DCA

Departmental administrators can be established by a Central Security Administrator, a ZCA, or a VCA, for a department that is linked to that VCA's division. The responsibilities that can be performed by a DCA include administrative tasks for the users and profiles that belong to this department.

In the following example, a VCA creates a DCA whose password expires when he first logs on; by specifying 15 for the second suboption instead of allowing it to default, CA-Top Secret prompts the user for a new password every fifteen days. For more information on passwords, refer to Chapter 5, *Passwords and System Entry Security*.

```
TSS CREATE(DCA1) NAME('Dave Dale') TYPE(DCA)
      PASS(BEST,15,EXP) DEPARTMENT(FINDEPT)
```

A department can have several DCAs or no DCAs. In the latter case, the administrative requirements for this department would have to be performed by either a Central Security Administrator or the appropriate VCA.

4.1.1.7 How To Create an Auditor

CA-Top Secret doesn't formally distinguish between an auditor and any other type of CA-Top Secret security administrator. Rather, auditors are implicitly defined by the functions they perform and the types of administrative authorities they are given.

As with any other type of administrator, an auditor's scope is a function of the TYPE it was designated when its ACID was created. In the following example, a divisional auditor is defined as a VCA whose password does not expire.

```
TSS CREATE(AUDV) NAME('Harry Hesse') TYPE(VCA)
      PASS(BEST,0) DIVISION(FINDIV)
TSS ADMIN(AUDV) ACID(AUDIT,REPORT) DATA(ALL,PROFILE) RES(INFO,REPORT)
      MISC8(LISTRDT,LISTSTC), MISC9(GENERIC)
```

4.1.2 Assigning Administrative Authority

An ACID's authority determines what he can do with respect to the administration of ACIDs, resources, facilities, displaying security database information, and so forth. An administrator can confer only those administrative authorities he already possesses.

There are nine levels of administrative authority. They are:

ACID	Indicates what functions the security administrator can perform on the ACIDs within his scope.
DATA	Designates which portions of a Security Record the security administrator can display when she issues a TSS LIST command function. For more information on the TSS LIST command function, refer to Chapter 4 in the section called: <i>Displaying Information</i> .
RESOURCE	Designates what type of maintenance a security administrator can perform on resources.
FACILITY	Governs over which facilities the security administrator has authority.
MISC1	Pertains to ACID access restrictions—such as LTIME, suspensions, and RDT maintenance. For more information on RDT maintenance, refer to Chapter 8 in the section called: <i>Maintaining the RDT</i> .
MISC2	Pertains to special facility-related resource administration.
MISC8	Enables the security administrator to list the contents of the RDT and FDT without the authority to maintain them, and can use the ASUSPEND administrative attribute.
MISC9	Addresses higher level administrative authority.
SCOPE	Determines the administrative scope of an LSCA.

Each of these levels is a keyword for the TSS ADMIN command function, and each keyword has designated operands which are used to further delineate administrative authority. You can assign one or more (or even all) of these different types, depending on your site requirements. The syntax for TSS ADMIN follows.

TSS ADMIN(acid) keyword(operand)

The *keyword* specifies the level of administrative authority, and the *operand* specifies the type of authority within that level. For example, the following command function allows the FINVCA Control ACID to display BASIC information—such as name, ACID type, access authorizations, system entry restrictions—for the ACIDs within his scope.

```
TSS ADMIN(FINVCA) DATA(BASIC)
```

In the next few sections, we will take a closer look at each level of administrative authority.

4.1.2.1 ACID Authorities

The ACID keyword of the TSS ADD command function is used to indicate what functions the security administrator can perform on the ACIDs within his administrative scope. The following example gives the FINVCA security administrator all the ACID authorities.

```
TSS ADMIN(FINVCA) ACID(ALL)
```

The operands used with this keyword are:

ALL	Authorizes the security administrator to perform all the authorities listed below. Note: The ALL operand is also used with the other keywords for administrative authority, and shouldn't be confused with the ALL Record.
AUDIT	Authorizes the security administrator to ADD or REMOVE the AUDIT attribute. (Used to monitor what an ACID does).
CREATE	Authorizes the security administrator to create and delete ACIDs.
DEFNODES	Authorizes the security administrator to designate an ACID's default command routing nodes for the Command Propagation Facility.
INFO	Authorizes the security administrator to use the TSS WHOHAS function.
MAINTAIN	Authorizes the security administrator to maintain information for currently existing ACIDs.
REPORT	Authorizes the security administrator to use the BATCH reporting utilities.
XAUTH	Authorizes the security administrator to permit one ACID to use another ACID for job submission.

4.1.2.2 DATA Authorities

The DATA keyword designates which portions of a Security Record the security administrator can display when he issues a TSS LIST command function. The following example gives the FINVCA security administrator the ability to display information concerning resources owned by an ACID.

```
TSS ADMIN(FINVCA) DATA(RESOURCE)
```

The operands used with the DATA keyword are:

ADMIN	Lists information about the ACID's administrative authorities.
ALL	Lists all information pertaining to an ACID except for password, profile, and session key.
BASIC	Lists general ACID information, such as name, type, and facility.
PCDATA	Lists information about ACID PC group data.
PASSWORD	Lists the ACID's password information (expiration date and interval).
PROFILE	Lists the profile(s) attached to the ACID.
PWVIEW	Lists the ACID's actual password as well as the expiration date and interval, if the PWVIEW control option is set to YES.
RESOURCE	Lists information about resources owned by an ACID.
SESSKEY	Lists the session key used to verify that one LU is authorized to link to another LU for the purposes of APPC conversation processing.
SOURCE	Lists information about the ACID's input device restrictions.
WORKATTR	Lists the SYSOUT delivery and accounting information associated with the ACID.
XAUTH	Lists resources permitted by an ACID within his scope, including information about what access levels are allowed and which ACID owns the resource.

To give an administrator all DATA administrative authorities, you must specify these options: ALL, PASSWORD or PWVIEW, SESSKEY (due to the sensitive nature of passwords and SESSKEY information), and PROFILE (since the profiles connected to an ACID may not necessarily fall under the scope of the security administrator). For example, in order to give the FINVCA security administrator the authority to list every possible item of Security Record information, you would have to issue the following command:

```
TSS ADMIN(FINVCA) DATA(ALL,PWVIEW,PROFILE,SESSKEY)
```

4.1.2.3 RESOURCE Authorities

The RESOURCE keyword is used to designate what type of maintenance a security administrator can perform on resources. The following example gives the FINVCA security administrator the ability to use BATCH reporting utilities for resources.

```
TSS ADMIN(FINVCA) RES(REPORT)
```

There are six operands that can be specified with the RESOURCE keyword. They are:

ALL	Allows the security administrator to perform all of the authorities listed below.
AUDIT	Allows the security administrator to ADD or REMOVE resources from the AUDIT record.
INFO	Allows the security administrator to issue TSS WHOHAS and TSS WHOOWNS for resources.
OWN	Allows the security administrator to define, move, and remove resource ownership.
REPORT	Allows the security administrator to use CA-Top Secret BATCH reporting utilities for resources.
XAUTH	Allows the security administrator to PERMIT or REVOKE resource access. The access levels must also be specified. In the following example, the command only allows FINVCA to PERMIT an access level of READ or FETCH for the ACIDs within his scope. TSS ADM(FINVCA) RES(XAUTH) ACCE(READ,FETCH)

Note: RES(XAUTH) is not a typical administrative authority, since it allows the administrator to assign or revoke access authorizations to any ACID (provided the resource involved is within his scope).

Specifying Resource With an Access Level:

When specifying RESOURCE, you must also specify an ACCESS level, if one applies (otherwise the default is READ). An example appears below.

```
TSS ADMIN(RESVCA) RES(OWN) ACCE(CREATE,UPDATE)
```

How to Assign Authority for One Resource Class:

The keyword, RESOURCE, applies to all resources. To indicate administrative authority for a specific resource, substitute that resource class name as shown below.

```
TSS ADMIN(RESVCA) PGM(INFO)
```

How to Administer a Resource Outside Your Scope:

The ACTION(ADMIN) keyword gives the security administrator the ability to allow ACIDs within his scope the authority to administer resources that aren't within it. In the following example, if the data sets with the high level index SYS1. aren't within the SCOPE of ACID USER05, the security administrator can issue the TSS PERMIT command function with ACTION(ADMIN) to allow USER05 to administer the data sets.

```
TSS PER(USER05) DSN(SYS1.) ACTION(ADMIN)
```

4.1.2.4 FACILITY Authorities

The FACILITY keyword determines over which facilities the security administrator has authority. The FACILITY keyword is used with the other administrative authorities as shown below.

```
TSS ADMIN(FINVCA) ACID(CREATE,MAINTAIN)
FACILITY(CICSTEST,CICSPROD)
```

This example gives the FINVCA security administrator the authority to create and maintain ACIDs for CICS users and profiles. Now the security administrator can give USER01 access to the CICS facilities as follows:

```
TSS ADD(USER01) FACILITY(CICSTEST,CICSPROD)
```

Note: CA-Top Secret doesn't check a security administrator's FACILITY authorization when he issues a TSS PERMIT allowing an ACID to access a resource only from a particular facility.

4.1.2.5 MISC1 Authorities

The keyword refers to the first of five levels of miscellaneous authorities. MISC1 sets authorities that generally pertain to ACID access restrictions—such as LTIME, suspensions, and RDT maintenance. The following example gives the FINVCA security administrator the ability to maintain and list the RDT and FDT Records.

```
TSS ADMIN(FINVCA) MISC1(RDT)
```

Use the following keywords to indicate what type of MISC1 authority a security administrator has.

- ALL** Allows the security administrator to perform all MISC1 type functions.
- INSTDATA** Allows the security administrator to associate installation data with ACIDs within his scope. It also authorizes administrators to ADD Dynamic Update Facility attributes, DUFEXTR, and DUFUPD, to ACIDs within his scope.
- LCF** Allows the security administrator to assign LCF (Limited Command Facility) command and transaction restrictions.
- LTIME** Allows the security administrator to maintain automatic terminal locktime intervals through the LTIME keyword. Issuing this command:

```
TSS ADMIN(FINVCA) MISC1(LTIME)
```

allows the FINVCA to administer the following command for USER01 (provided USER01 is in FINVCA's scope of authority) which causes USER01's terminal to automatically "lock" after a 30 minute interval of inactivity.

```
TSS ADD(USER01) LTIME(30)
```

- Note:** The security administrator's ACID must also possess ACID(MAINTAIN) authority to assign the LTIME parameter.
- NOATS** Allows the security administrator to prevent ACIDs (within his scope) from signing on through Automatic Terminal Signon (ATS). This applies to the CICS, IMS, and CA-IDMS facilities.
- RDT** Allows the security administrator to maintain and list the RDT and FDT Records.

SUSPEND	Allows the security administrator to suspend and unsuspend ACIDs by using the SUSPEND keyword with the ADD or REMOVE function.
TSSSIM	Allows the security administrator to use the TSSSIM utility for testing and diagnostic purposes.
USER	Allows the security administrator to administer the use of unownable installation-defined resources (USERx).

4.1.2.6 MISC2 Authorities

The MISC2 keyword refers to the second of five levels of miscellaneous authorities. MISC2 defines authorities pertaining to special facility-oriented resource administration. The following example gives the FINVCA security administrator the ability to assign TSO-related data fields to an ACID.

```
TSS ADMIN(FINVCA) MISC2(TSO)
```

Use the following operands to indicate what type of MISC2 authority a security administrator has.

APPCLU	Allows the security administrator to maintain the APPCLU Record and assign APPC-related fields to an ACID.
DLF	Allows the security administrator to assign data sets from the DLF (Data Lookaside Facility) Record.
NDT	Allows the security administrator to assign VAX-related data fields to an ACID.
PC	Allows the security administrator to assign PC group attributes.
SMS	Allows the security administrator to assign SMS-related data fields to an ACID.
TARGET	Allows the security administrator to assign nodes associated with ACIDs.
TSO	Allows the security administrator to assign TSO-related data fields to an ACID.
WORKATTR	Allows the security administrator to assign accounting and SYSOUT delivery information fields to an ACID.

4.1.2.7 MISC3 Authorities

The MISC3 keyword refers to the third of five levels of miscellaneous authorities. MISC3 defines authorities pertaining to the creation and maintenance of the Static Data Table (SDT). The following example gives the FINVCA security administrator the ability to define records for the Static Data Table (SDT).

```
TSS ADMIN(FINVCA) MISC3(SDT)
```

You can also restrict a security administrator to only listing the contents of the SDT by using MISC8(LISTSDT). See the next section for details.

4.1.2.8 MISC8 Authorities

The MISC8 keyword refers to the fourth of five levels of miscellaneous authorities. This particular keyword allows administrators to list the Resource Descriptor Table (RDT), Field Definition Table (FDT), and the Started Task Table **without** being given the authority to maintain the them. This authority allows for a separation of responsibilities between viewing the tables and modifying them. MISC8 authority is also used to authorize use of the ASUSPEND keyword to remove administrative suspensions.

The following example gives the FINVCA security administrator the ability to administer Multiple Console System (MCS) commands.

```
TSS ADMIN(FINVCA) MISC8(MCS)
```

Use the following operands to indicate what type of MISC8 authority the security administrator has.

LISTRDT	Allows the security administrator to list the Resource Descriptor Table and the Field Definition Table.
LISTSTC	Allows the security administrator to list the Started Task Table.
LISTSDT	Allows the security administrator to list the record elements in the Static Data Table (SDT).
MSC	Allows the security administrator to issue Multiple Console System (MCS) commands for ACIDs within their scope.
REMASUSP	Allows the security administrator to issue the TSS REMOVE(acid) ASUSPEND command function that removes administrative suspensions.

4.1.2.9 MISC9 Authorities

The final miscellaneous authority keyword—MISC9—deals with higher level administrative authority. The following example gives the FINVCA security administrator the ability to issue a security trace on a user's activities.

```
TSS ADMIN(FINVCA) MISC9(TRACE)
```

Use the following operands to indicate what type of MISC9 authority the security administrator has.

ALL	Allows the security administrator to perform all of the MISC9 type functions listed below.
BYPASS	Allows the security administrator to assign any of the various "security-bypass" attributes (also called "no-check" attributes).
CONSOLE	Allows the security administrator to assign the CONSOLE attribute to any ACID within his scope.
GENERIC	Allows the security administrator to issue generic WHOOWNS inquiries to obtain a list of all resources owned within his scope.
GLOBAL	Allows the security administrator to PERMIT or REVOKE access to the ALL Record.
MASTFAC	Allows the security administrator to associate a region control ACID with a multiuser address space facility.
MODE	Allows the security administrator to specify security modes for users.
STC	Allows the MSCA/SCA to define a started task to the CA-Top Secret Started Task Table, and authorizes the administrator to list the contents of the Started Task Table.
TRACE	Allows the security administrator to ADD or REMOVE the TRACE attribute which sets a security trace on a user's activities.

4.1.2.10 SCOPE Authority

SCOPE authority is used by the MSCA to determine the administrative scope of an LSCA (the only Control ACID type with variable scope). That scope can include Zones and even other LSCAs. For example, the command shown below designates LSCA02, LSCA03, and ZONE1 as being within the administrative scope of LSCA01.

```
TSS ADMIN(LSCA01) SCOPE(LSCA02,LSCA03,ZONE1)
```

LSCA01 now has scope over ZONE1, LSCA02, and LSCA03, and over everything within the scope of LSCA02 and LSCA03. SCOPE authority does not apply to any other Control ACID and cannot be defined by any Control ACID other than the MSCA.

4.1.3 Removing Administrative Authorities

The DEADMIN function is used to remove administrative authorities. All formats, rules, and restrictions that apply to the ADMIN function are also applicable to the DEADMIN function.

With the following syntax, the *keyword* specifies the level of administrative authority, and the *operand* specifies the type of authority within that level.

```
TSS DEADMIN(acid) keyword(operand)
```

For example, the following command function removes the authority from the FINVCA Control ACID to display BASIC information—such as name, ACID type, access authorizations, system entry restrictions—for the ACIDs within his scope.

```
TSS DEADMIN(FINVCA) DATA(BASIC)
```

For a detailed explanation of all of the administrative keywords that can be used with the DEADMIN function, refer to the *Command Functions Guide*.

4.2 Identifying Users by Defining ACIDs

CA-Top Secret controls access to your system by first identifying users through an ACcessor ID, also known as an ACID.

How to Create an ACID: The TSS CREATE command function is used to define new ACIDs to CA-Top Secret, thereby establishing a Security Record for that ACID. Many different parameters can be associated with the TSS CREATE command function.

Users can be established by any administrator with ACID(CREATE) administrative authority. While a user can be assigned most types of administrative authority, the user's scope is always limited to himself. In general, it is advisable to allow a user minimal administrative authority, leaving administrative functions in the hands of Control ACIDs.

However, one administrative authority that is often given to users is the ability to obtain BASIC information about themselves. In this case, DATA type authority is required. For example:

```
TSS ADMIN(USER01) DATA(BASIC)
```

gives USER01 the ability to display her basic Security Record information (except for password, profile, and session information) by entering:

```
TSS LIST(USER01) DATA(ALL)
```

ACIDs are the building blocks of your security hierarchy—they identify authorized users, and their designated access rights and restrictions to CA-Top Secret. Because of their importance in the CA-Top Secret security environment, ACIDs are protected in various ways. Before they can be explicitly protected, however, they must first be defined to the system.

The 4.5, “ACID Creation Requirements Chart” on page 4-35 gives a synopsis of the required and optional parameters used to create different types of ACIDs.

4.2.1 Adding Department, Division, and Zone ACIDs

To define a zone, division, or department, only two parameters are required in addition to the eight-character ACID.

TYPE Identifies whether you are creating a department, division, or zone.

NAME Identifies its name. Names can be up to 32 characters in length, must be surrounded by single quotes if embedded with blanks, and can use letters, numbers, and special characters.

Password is invalid with such organizational ACIDs.

The general syntax for such organizational ACIDs is:

```
TSS CRE(acid) TYPE(type) NAME('name-of-organizational-acid')
```

The following sections illustrate in more detail how to create a zone, division, and department.

4.2.1.1 How to Create a Zone

A Zone ACID is another optional organizational level in your corporate structure that can be used to group two or more divisions in your CA-Top Secret security database. A Department ACID can't be directly attached to a Zone, but can be attached to a Division ACID that is attached to a Zone ACID. Resources can be assigned to Zones, but it is not recommended.

To define a zone the following parameters are required:

TYPE Identifies that you are creating a zone.

NAME Identifies its name. Names can be up to 32 characters in length, must be surrounded by single quotes if embedded with blanks, and can use letters, numbers, and special characters.

In the following example, CA-Top Secret creates a Parts Zone.

```
TSS CRE(PARTZON) TYPE(ZON) NAME('PARTS ZONE')
```

A zone can have one or more ZCA administrative ACIDs assigned to administer various authorities for other ACIDs assigned to the zone.

4.2.1.2 How to Create a Division

CA-Top Secret allows you to optionally define multiple divisions within your corporate structure. Each division can comprise one or more departments, and is assigned a unique Division ACID. Resources can be assigned to a division, but it is not recommended.

To define a division the following parameters are required:

TYPE Identifies that you are creating a division.

NAME Identifies its name. Names can be up to 32 characters in length, must be surrounded by single quotes if embedded with blanks, and can use letters, numbers, and special characters.

Optionally, you can enter:

ZONE Indicates the zone to which the division belongs.

The following example creates a new Accounting Division that is placed in the Parts Zone.

```
TSS CRE(ACCTDIV) TYPE(DIV) NAME('ACCT DIV')
      ZON(PARTZON)
```

A division can have one or more VCA administrative ACIDs assigned to administer various authorities for other ACIDs assigned to the division.

4.2.1.3 How to Create a Department

At any installation users typically work for a particular department. CA-Top Secret recognizes this logical separation and associates User ACIDs with departments. Every department is assigned a unique Department ACID. Resources can be assigned to a department, which is the recommended location.

To define a department the following parameters are required:

TYPE Identifies that you are creating a department.

NAME Identifies its name. Names can be up to 32 characters in length, must be surrounded by single quotes if embedded with blanks, and can use letters, numbers, and special characters.

DIVISION Identifies the division to which the department belongs. This is only required if a department is associated with a particular division.

In the following example, CA-Top Secret creates the Personnel Applications Department and links it to the Accounting Division.

```
TSS CRE(ACCTDEPT) TYPE(DEPT) NAME('PERSONNEL APPL')
      DIV(ACCT01)
```

Note: A Department ACID can't be directly attached to a Zone ACID. It must be attached to a Division ACID which is attached to a Zone ACID.

4.2.2 Creating User ACIDs

The following parameters must be incorporated into the TSS CREATE command function when a new user is defined to CA-Top Secret:

NAME Identifies the user's name. Names can be up to 32 characters in length, must be surrounded by single quotes if embedded with blanks, and can use letters, numbers, and special characters.

PASSWORD Specifies an up to eight-character password.

DEPARTMENT

Indicates the department to which the user belongs. This keyword is required if the Control ACID issuing the command is an SCA, LSCA, or ZCA, but not a DCA.

Optionally, you can enter:

TYPE Specifies the USER. It isn't necessary since USER is the default.

The following example defines a new TSO user attached to the PAYDEPT department. The password INITIAL expires as soon as he logs on; he will also be prompted for a new one that will last for seven days.

```
TSS CRE(USER02) NAME('ANDY POE')
      DEPT(PAYDEPT)
      PASS(INITIAL,7,EXPIRED) FAC(TSO)
```

to enter the commands in the previous example, the CA-Top Secret security administrator must possess ACID(CREATE) administrative authority and a scope that encompasses PAYDEPT.

4.2.3 Using an Existing ACID to Create Another

Security administrators can use a model ACID to create a large number of new ACIDs. All BASIC information is copied from the model or *template* ACID, and the security administrator can also change, add, or omit any BASIC information from the model. BASIC information includes everything that is listed when issuing a TSS LIST(acid) DATA(BASIC) command function. This information includes: the name associated with the ACID, the ACID type, facilities permitted access, associated profiles, and when the ACID was created, modified, and last used. Any existing ACID can be used as the model, but the model must exist **before** attempting to create copies from it. All ACID types can be created using the syntax shown below.

```
TSS CREATE(newacid) USING(modelacid)
```

All rules of scope and administrative authority are supported. If the model ACID is outside the scope of the security administrator and/or the model ACID has any information field that requires an ADMIN authority the security administrator doesn't have, the model ACID can't be used to create the new ACID. For example, a security administrator needs MISC1(INSTDATA) authority to copy INSTDATA information from the model ACID to the new ACID. If he doesn't have this particular authority, the CREATE function will **not** be successful.

In the following example, a new ACID called USER03 will be created using the ACID USER02 as the model.

```
TSS CREATE(USER03) USING(USER02)
```

USER03 now possesses all of the BASIC user information of USER02—including the name and password.

How to Change BASIC Information: To give USER03 a name and password different from the model, USER02, enter:

```
TSS CREATE(USER03) USING(USER02)
      NAME('JOHN SMITH') PAS(WELL,7,EXP)
```

Use the same process for any other information field that needs changing from the model ACID, (all of the rules of scope and administrative authority apply). For example, to change the name and password of USER03, and the contents of the INSTDATA field (since it contains USER02's date of hire) and to copy all of the other BASIC information fields belonging to USER02, enter the command shown next.

```
TSS CREATE(USER03) USING(USER02)
      NAME('JOHN SMITH') PAS(WELL,7,EXP)
      INSTDATA('MARCH 85')
```

How to Omit BASIC Information:

If you didn't want to include any INSTDATA information when using USER02 as the model ACID, you would enter:

```
TSS CREATE(USER03) USING(USER02)
      NAME('JOHN SMITH') PAS(WELL,7,EXP)
      INSTDATA( )
```

This technique of omitting certain items can be used for all information **except** attributes because they have no values to nullify within the parentheses. To remove an attribute, such as NODSNCHK, use the REMOVE function after creating the ACID.

4.2.4 Promoting or Demoting ACIDs

Given the proper administrative authority, you can enter the MOVE function to move an ACID from one department, division or zone to another, or to change a user, ZCA, DCA, and/or VCA into an SCA.

The following considerations apply when performing a MOVE:

- Only the MSCA can move a USER to an SCA or LSCA
- Only the MSCA can move an LSCA or SCA
- Only the MSCA, SCA or ZCA (within scope) can move a DIV or VCA
- Only the MSCA, SCA, ZCA, or VCA (within scope) can move a DEPT or DCA
- Only the MSCA, SCA, ZCA, or DCA (within scope) can move a USER or PROFILE

If you use TYPE, you can promote or demote the type of ACID. As the examples that follow illustrate, you can promote a user to a DCA, or demote an SCA to a user.

How to Promote a User to a DCA: The following command, issued by a VCA, would leave a user in the same department but make him or her a DCA.

```
TSS MOVE(USER04) TYPE(DCA)
```

Before the MOVE function, USER04 was a user ACID. After the move, USER04 now belongs to the same department but is promoted to a DCA.

How to Demote a DCA to a User: With the TSS MOVE command function, a VCA, ZCA, LSCA, or SCA can move a DCA into a new department and demote it to a TYPE(USER).

```
TSS MOVE(DCA01) DEPT(PAYDEPT) TYPE(USER)
```

Before the MOVE function, DCA01 belonged to the PERDEPT department as a DCA. After the move, DCA01 belongs to the PAYDEPT department as a user.

How to Demote an SCA to a User: An SCA can become a TYPE(USER) by simply issuing the target department.

```
TSS MOVE(SCA01) DEPT(FINDEPT) TYPE(USER)
```

Before the MOVE function, SCA01 was an SCA. After the move, SCA01 now belongs to department FINDEPT as a user ACID.

4.2.5 Deleting ACIDs

The DELETE function purges the designated ACID from the CA-Top Secret Security File. This example deletes the ACID USER999.

```
TSS DELETE(USER999)
```

All authorizations granted to USER999 are automatically revoked and resources owned by USER999 are automatically freed.

Note these restrictions when using DELETE:

- If the ACID still owns resources that have been permitted to other users, you can't delete it.
- If the ACID is still permitted to other ACIDs (that is, job submit authority), it can't be deleted.
- A divisional, departmental, or profile ACID that still has other ACIDs linked to it can't be deleted.
- The ALL, AUDIT, DLF, FDT, NDT, RDT, and STC Records and the MSCA's ACID can never be deleted.

4.2.6 Setting ACID Expiration Dates

When used with either the CREATE or ADD function, the FOR or UNTIL parameters set time limits on how long a user ACID can be used before it becomes invalid.

The FOR parameter specifies how many days an ACID will be valid, starting with the day on which the FOR parameter is entered. The command shown below would limit USER01's access to the system to just that day.

```
TSS ADD(USER01) FOR(1)
```

The UNTIL parameter instructs CA-Top Secret to automatically expire the designated ACID on the indicated date. This command allows the ACID USER01 access until December 31, 1999.

```
TSS ADD(USER01) UNTIL(12/31/99)
```

Note: Both of these parameters can be used with the PERMIT function to specify a time limit on authorized access to the designated resource(s).

4.2.6.1 Deactivating an Expiration Date to Reactivate a User

There are five commands to reactivate ACIDS that have expired. The first two use the FOR parameter which uses a day value as its operand, the next two use the UNTIL which uses a date value, and the last one uses the EXPIRE attribute.

Either parameter of the two commands below sets the FOR parameter to zero expiration days, thereby reactivating the ACID.

```
TSS REM(USER01) FOR(0)
```

```
TSS ADD(USER01) FOR(0)
```

Or, either of the commands below, sets the UNTIL parameter to no specified date value, thereby reactivating the ACID.

```
TSS REM(USER01) UNTIL
```

```
TSS ADD(USER01) UNTIL
```

4.2.6.2 Changing an Expiration Date

To change the expiration date of an ACID, use the REPLACE function with the FOR or UNTIL parameter as shown below.

```
TSS REP(USER01) UNTIL(03/31/95)
TSS REP(USER01) FOR(30)
```

If you want to remove an expiration date and are unsure of whether you specified FOR or UNTIL on the TSS ADD command function, you can use the syntax shown below.

```
TSS REM(USER01) EXPIRE
```

4.2.6.3 Setting Automatic Inactivity Suspensions

To prohibit ACIDs that haven't been used for long periods of time, CA-Top Secret provides the INACTIVE control option. Inactivity is measured from the day that an ACID's password expires. If CA-Top Secret does not detect any activity for this ACID before the threshold is reached, the ACID will be suspended. A system-wide threshold varying anywhere from 0 to 255 days can be selected as the period of inactivity that causes the automatic suspension of the ACID.

Note: Changing the password is considered activity.

Reactivating an ACID: A CA-Top Secret security administrator with ACID(MAINTAIN) authority and the appropriate scope can reactivate a suspended ACID by removing the SUSPEND attribute from the user and replacing the expired password as shown below.

```
TSS REM(USER01) SUSPEND
TSS ADD(USER01) PASS(QUACK,,EXP)
```

4.2.6.4 Setting Limited Duration Suspensions

To suspend an ACID until a specified date, use the SUSPEND function with the UNTIL parameter. This feature simplifies administering the temporary deactivation of accounts during vacations, extended trips, leaves of absences, and so on. In the following example, the ACID will then be automatically reactivated on January 5, 1999.

4.2 Identifying Users by Defining ACIDs

For example:

```
TSS ADD(USER01) SUSPEND UNTIL(01/05/99)
```

4.3 Creating Profiles

A profile is an ACID of resource permissions allowing a collection of users who can share access authorizations for protected resources, and is another way to minimize the use of access authorizations made directly to users. Use of profiles avoids redundant authorizations and should make the administration of your security database much easier. Define profiles for the access permissions required to perform a specific function, then assign the profile to whoever needs the permissions. Whenever another user needs to perform that function, the profile can simply be added to that user's ACID. Remember the following points when you are creating and using profiles:

- You can attach a maximum of 254 profiles to one user.
- You can't assign a password to a profile.
- You can't attach a profile to another profile. A profile must be defined under a department, or a division, or a zone attached to a user.
- To attach a profile to a user, **both** the ACID and the profile must be within your scope of authority.
- Profiles can be made globally administrable with the GAP attribute.
- Profiles should not own resources.
- Users can be assigned to a profile permanently or temporarily.

How to Define a Profile:

To define a new profile, follow this basic procedure:

Step 1 Assign an owner using the following required parameters with the CREATE function:

TYPE	Specifies PROF as the type of ACID you want to create.
NAME	Identifies its name. Names can be up to 32 characters in length, must be surrounded by single quotes if embedded with blanks, and can use letters, numbers, and special characters.

DEPARTMENT

Indicates the department to which the profile belongs. This keyword is required if the Control ACID issuing the command is an SCA, LSCA, or ZCA, but not a DCA.

The following example creates the new Payroll Department profile which is linked to that department.

```
TSS CRE (APPPRO1) TYPE (PROF) NAME ('PAYROLL DEPT PROF')
      DEPT (PAYROL)
```

Step 2 Authorize the profile's resource access using the the PERMIT function.

In the following example profile APPPRO1 is granted update access to all data sets beginning with ABC as the high level qualifier from 8 a.m. to 7:59 p.m.

```
TSS PERMIT (APPPRO1) DSN (ABC.) ACC (UPDATE) TIME (08,20)
```

How to Attach a Profile to an ACID:

Use either the CREATE or the ADD function to attach a profile to an ACID. The following example shows how to add the authorizations granted to APPPRO1, created above, to USER01.

```
TSS ADD (USER01) PROFILE (APPPRO1)
```

To attach a profile to an ACID, you must possess ACID(MAINTAIN) authority and both the profile and the ACID must lie within your scope—unless it is a globally administrable profile. For more details, refer to How to Create a Global Profile on page 4-31.

How to Attach a Profile Temporarily: Profiles can be added to a user ACID on a temporary basis using the FOR or UNTIL parameters. For example, issue the command shown below to connect the profile ACID to the specified user ACID for the next ten days. After that time, CA-Top Secret automatically removes the profile from the ACID.

```
TSS ADD (USER01) PROFILE (APPPRO01) FOR (10)
```

How to List an Expiration Date for a Profile:

To list the expiration date for a temporary profile, use the EXPIRE operand with the DATA keyword when issuing the TSS LIST command function as shown below.

```
TSS LIST (APPPRO01) DATA (EXPIRE)
```


How to Order Profiles: Because the order of profiles directly affects resource checking of the permissions granted, the ADD function can designate where in the order a **new** profile is added.

Note: Existing profiles can't be reordered, only new ones being added.

The default always adds the new profile at the end of the list of profiles. To put the new profile somewhere other than at the end, use the syntax shown below.

```
TSS ADD(acid) PROFILE(newprof) BEFORE|AFTER(existingprof)
```

For example, an ACID called USER04 currently has four profiles: PROF1, PROF2, PROF3, and PROF4. To add a new profile called PROF5 that should be added after PROF1 in the search order, issue one of the commands below.

```
TSS ADD(USER04) PROFILE(PROF5) AFTER(PROF1)
or
TSS ADD(USER04) PROFILE(PROF5) BEFORE(PROF2)
```

You can add an existing profile and make it the first in the search order by using the **FIRST** keyword.

For example, to add an existing profile called PROF4 to USER04 and make it the first in the search order, issue the command below.

```
TSS ADD(USER04) PROFILE(PROF4) FIRST
```

How to Create a Global Profile: A conventional department profile can only be attached to users if both the user and the profile fall within the administrator's scope. In some cases, however, it is desirable to create a profile that might be available to all administrators. Such a profile is called a *globally administrable profile*, or GAP. Only a security administrator, with the same scope of authority as the security administrator who created the profile, can assign the GAP attribute. A profile allows any CA-Top Secret security administrator to attach it to any user within his administrative scope.

To create a globally administrable profile, add the GAP attribute as shown below.

```
TSS ADD(PROF01) GAP
```

Now any administrator can attach PROF01 to any user within her scope.

How to Remove the Global Attribute: To remove the GAP attribute from a profile use the REMOVE function as shown below.

```
TSS REM(PROF01) GAP
```

4.4 Creating Groups

A group is similar to a profile in that it is a collection of users. The difference between profiles and groups is that groups are recognized by IBM OpenEditionMVS.

Each User ACID must be associated with at least one group if that particular group is recognized by IBM OpenEditionMVS.

Remember the following points when you are creating and using groups:

- You can attach a maximum of 254 groups and profiles to one user.
- You can't assign a password to a group.
- You can't attach a group to another group. A group must be attached to a department and a user.
- To attach a group to a user, **both** the ACID and the group must be within your scope of authority.
- Groups cannot own resources.
- Users can be assigned to a group permanently or temporarily.

How to Define a Group: A Group type ACID can have DFP and GID information attached to it.

To define a new group, follow this basic procedure: Assign an owner using the following required parameters with the CREATE function:

TYPE	Specifies GROUP as the type of ACID you want to create.
NAME	Identifies its name. Names can be up to 32 characters in length, must be surrounded by single quotes if embedded with blanks, and can use letters, numbers, and special characters.
DEPARTMENT	Indicates the department to which the group belongs. This keyword is required if the Control ACID issuing the command is an SCA, LSCA, or ZCA, but not a DCA.

The following example creates the new Payroll Department group which is linked to that department.

```
TSS CRE(APPGR01) TYPE(GROUP) NAME('PAYROLL DEPT PROF')
      DEPT(PAYROL)
```

In the following example group APPGR01 is granted update access to all data sets beginning with ABC as the high level qualifier from 8 a.m. to 7:59 p.m.

```
TSS PERMIT(APPGR01) DSN(ABC.) ACC(UPDATE) TIME(08,19)
```

How to Attach a Group to an ACID: Use either the CREATE or the ADD function to attach a group to an ACID. The following example shows how to add the authorizations granted to APPGR01, created above, to USER01.

```
TSS ADD(USER01) GROUP(APPGR01)
```

To attach a group to an ACID, you must possess ACID(MAINTAIN) authority and both the group and the ACID must lie within your scope.

How to Attach a Group Temporarily: Groups can be added to a user ACID on a temporary basis using the FOR or UNTIL parameters. For example, issue the command shown below to connect the group ACID to the specified user ACID for the next ten days. After that time, CA-Top Secret automatically removes the group from the ACID.

```
TSS ADD(USER01) GROUP(APPGR001) FOR(10)
```

How to List an Expiration Date for a Group: To list an expiration date for a temporary group, use the EXPIRE operand with the DATA keyword when issuing the TSS LIST command function as shown below.

```
TSS LIST(APPGR001) DATA(EXPIRE)
```

4.5 ACID Creation Requirements Chart

The following table provides, in a quick reference format, the parameters that are either required or generally indicated when the various types of ACIDs are created. Optional parameters are shown in brackets.

Table 4-1. ACID Creation Chart	
Type of ACID	Model TSS CREATE Entry
SCA	TYPE(SCA) NAME('name') PASSWORD(password) [FACILITY(facilities)]
LSCA	TYPE(LSCA) NAME('name') PASSWORD(password) [FACILITY(facilities)]
ZCA	TYPE(ZCA) NAME('name') ZONE(zone) PASSWORD(password) [FACILITY(facilities)]
ZONE	TYPE(ZONE) NAME('name')
DIVISION	TYPE(DIVISION) NAME('name') [ZONE(zone)]
VCA	TYPE(VCA) NAME('name') DIVISION(division) PASSWORD(password) [FACILITY(facilities)]
DEPARTMENT	TYPE(DEPT) NAME('name') [DIVISION(division)]
DCA	TYPE(DCA) NAME('name') DEPT(department) PASSWORD(password) [FACILITY(facilities)]
GROUP	TYPE(GROUP) NAME('name') DEPT(department)
PROFILE	TYPE(PROFILE) NAME('name') DEPT(department)
USER	NAME('name') DEPT(department) PASSWORD(password) [FACILITY(facilities)]

4.6 Displaying Information

One of the primary ways CA-Top Secret aids the security administrator is by providing online information regarding the status of the security environment and the ACIDs and resources within that environment. The type of information displayed depends on which command is used.

There are five "informational" TSS commands. They are:

TSS LIST	Displays information on individual as well as reserved ACIDs—such as the RDT.
TSS WHOHAS	Displays information on a resource or ACID depending on which is specified. If a resource, provides information on the resource owner and the ACIDs authorized to access it; if an ACID, provides information on additional ACIDs to submit jobs for the specified ACID.
TSS WHOOWNS	Displays information on whether a resource is defined to CA-Top Secret and who is the owner.
TSS WHOAMI	Displays information on the security environment for the ACID currently signed on to the terminal.
TSS MODIFY(STATUS)	Displays information on the global security environment and control options—such as the default MODE—currently in effect.

In this chapter, we will take a close look at the commands needed to display information about the security database, including sample output for each.

4.6.1 Displaying ACID and Record Information

Use the TSS LIST command function to display information contained on individual ACIDs, the ALL Record, the RDT Record, the FDT Record, the DLF Record, the NDT Record, the APPCLU Record, or the Audit Record. The information listed depends on two factors:

- the scope of the ACID issuing the TSS LIST command,
- the administrative authorities he possesses

For example, to list information on:

This Record Type:	You would need:
Security Record	DATA (options) authority needed
ALL Record	MISC9(GLOBAL) authority needed
Audit Record	RESOURCE(AUDIT) authority needed

Only the MSCA or an authorized security administrator can display installation-wide information using the TSS LIST command function. The administrative authorities needed are shown below.

```
TSS ADMIN(ALL) DATA(ALL,PWVIEW,PROFILE,SESSKEY)
```

You can, however, grant every ACID the authority to list the information stored in his own Security Record (with the exception of password, profile, and session key information), by issuing the following TSS ADMIN command:

```
TSS ADMIN(ALL) DATA(ALL)
```

That way, if FINVCA entered TSS LIST(FINVCA) DATA(ALL), CA-Top Secret would display the following information.

```

ACCESSORID   = FINVCA   NAME       = JOHN Q PUBLIC
TYPE         = DIV C/A  SIZE       =      2816 BYTES
FACILITY     = BATCH
FACILITY     = STC
FACILITY     = TSO
FACILITY     = CICSPROD
FACILITY     = VSEA
FACILITY     = VM
CREATED      = 01/01/96 LAST MOD   = 02/15/97
PROFILES     = TCSPROF
LAST USED    = 10/03/96 10:27 CPU(XA81) FAC(VM) COUNT(00002)
XAUTH DSN    = CAI.TSSDOC.DLF      OWNER(DEVLDEPT)
  ACCESS     = UPDATE
XAUTH CPCMD  = TRANSFER            OWNER(PSGDEPT)
  ACTION     = VMPRIV
XAUTH MDISK  = MAINT.0193          OWNER(PSGREH)
  ACCESS     = READ
  ACTION     = PASSWORD
-----ADMINISTRATION AUTHORITIES
RESOURCES    = *ALL*
  ACCESS     = ALL
ACID         = *ALL*
LIST DATA   = *ALL*,PROFILES,PASSWORD
MISC9        = TRACE,GLOBAL,GENERIC

TSS0300I LIST   FUNCTION SUCCESSFUL

```

As you can see by this example, the TSS LIST command displays some useful information, including:

- ACID holder's name (JOHN Q. PUBLIC)
- ACID type (Divisional Control ACID or VCA)
- Size of the Security Record
- Facilities the ACID can access (BATCH, STC, TSO, CICSPROD, VSEA, and VM)
- When the ACID was created (1/1/94), last modified (2/15/95) and last used (10/3/94)
- What resources are permitted to the ACID (a CP command, a minidisk, and a data set)

The listing also includes the administrative authorities that FINVCA has been granted and concludes with a CA-Top Secret message indicating that TSS LIST worked successfully.

How to List Resources Within Scope: To find out what resources in your scope of authority are being audited, you would enter this TSS command:

```
TSS LIST(AUDIT)
```

How to List Users:

To list the ACIDs and associated names in a department within your scope of authority, you would enter this TSS command:

```
TSS LIST(ACIDS) DATA(NAMES) DEPARTMENT(DEPT01)
```

4.6.2 Displaying Access Authorization Information

The information provided by the TSS WHOHAS command function depends on the operand supplied.

- When a **resource** is specified, CA-Top Secret provides information about the resource's owner and the ACIDs authorized to access it. This kind of function requires RESOURCE(INFO) authority.
- When an ACID is specified as the operand, CA-Top Secret provides a list of which additional ACIDs are authorized to submit jobs for that ACID. This kind of function requires ACID(INFO) authority.

How to Determine Resource Owner and Authorized ACIDs:

To determine the owner and all the ACIDs authorized to use the VSE. data set, enter:

```
TSS WHOHAS DSN(VSE.)
```

generating this kind of display:

```
TSS WHOHAS DSN(VSE.)
DATASET   = VSE.                OWNER(DSNDEPT)
XAUTH     = VSE.SYSLIB          ACID(FUNUSER  )
ACCESS    = READ
FAC       = BATCH
TSS0300I WHOHAS    FUNCTION SUCCESSFUL
```

As this sample listing shows, the owner of the VSE. data set resource is the DSNDEPT ACID, and FUNUSER has been authorized to use VSE.SYSLIB with READ access on BATCH.

Note: You can also use one of the WHOHAS-DATA operands to indicate exactly how CA-Top Secret should interpret the resource being specified (that is, Is it a prefix? Is a mask being used? Should it be taken at face value?).

How to Determine Owner and Submitting Job ACIDs: To determine the owner and all the ACIDs authorized to submit jobs for that ACID, enter:

```
TSS WHOHAS ACID(DB2US12)
```

generating this kind of display:

```
TSS WHOHAS ACID(DB2US12)
ACID      = DB2US12          OWNER(DB2US12)
XAUTH     = DB2US12          ACID(DB2SCA1  )
TSS0300I WHOHAS    FUNCTION SUCCESSFUL
```

As this sample listing shows, the owner is DB2US12 and the DB2SCA1 is authorized to submit jobs for this owning ACID.

4.6.3 Displaying Resource Ownership

The TSS WHOOWNS command function is used to determine if a resource is defined to CA-Top Secret and who the owner is. All resources that generically match the designated resource will be displayed in alphabetical order.

As with the previous commands, the information provided by TSS WHOOWNS is delimited by the scope of the administrator issuing the command. The required authority for its use is RESOURCE(INFO).

How to Display All Owned Resources: To display all the owned resources for a particular resource class, you would use an asterisk (*) as the operand. For example, to determine all the owned transactions within your scope, enter:

```
TSS WHOOWNS OTRAN(*)
```

which would generate this kind of display:

```
BRER004 OWNS OTRAN ABC
DABAD01 OWNS OTRAN TCAI
TSS0300I WHOOWNS FUNCTION SUCCESSFUL
```

4.6.4 Displaying Individual ACID Security Information

Unlike the other TSS commands, the TSS WHOAMI command can be used by any user. When TSS WHOAMI is entered at an online terminal, CA-Top Secret displays security environment information for the ACID that is currently signed on to that terminal.

How to Display the Security Environment for an ACID: The following sample display demonstrates the type of information provided in response to a TSS WHOAMI command:

```
TSS WHOAMI
TSS0303I Acidname(LCLADMIN) Type(SCA ) Mode(WARN)
TSS0303I Facility(VMSYSC ) Terminal(L1036132)
TSS0303I Systemid(SYSTEMC ) Logid(SYSC) Log(INIT,MSG)
```

As the example shows, TSS WHOAMI provides useful information about who is signed on to that terminal—such as his ACID, and ACID type, the terminal ID, and facility currently logged on to, and associated logging options.

4.6.5 Displaying the Global Security Environment

The TSS MODIFY(STATUS) command function can be entered by any ACID—usually the security administrator—with CONSOLE attribute. This command issues a display indicating the global security environment and control options—such as the default security MODE, default LOG, and PASSWORD options—that are currently in effect.

How to Display the Global Security Environment: The example on the following page shows the kinds of information provided for a TSS MODIFY(STATUS) command.

4.6 Displaying Information

```
TSS MODIFY(STATUS)
TSS MODIFY(STATUS)
TSS9661I CA-Top Secret BASE      Status
AUTH(OVERRIDE,ALLOVER)  ADMINBY(NO)          CACHE(ON)
AUDIT FILE(065%)        RECOVERY FILE(000%)  SECURITY FILE(030%)
ID=PRIMARY              SHRFILE(YES)
BACKUP(ACTIVE-01:00)    Last change on 03/08099 at 05:14
AUTOERASE(NO)           CANCEL(NO)            DATE(MM/DD/YY)
DEBUG(OFF)              DIAGTRAP(OFF)         SECTRACE(OFF)
IOTRACE(OFF)           SFSDIAG(OFF)         GTRACE(OFF)
DOWN(BW,SB,TW,OW)      EXIT(OFF)             EXPDAYS(00)
LOG(SEC9,MSG)          LOGBUFF(0032)         CURRBUF(0004)
MODE(FAIL)              SWAP(NO)              SYSOUT(A,LOCAL)
OPTIONALS(NONE)
TAPE(OFF)               TEMPDS(NO)            TIMER(030)
TIMELOCK(050,032,064,064) MFACESS(DORM)        INACTIVE(030)
CMDNUM(02)              VTHRESH(005,NOT)
STATUS(BASE,JES,PASSWORD,FACMODE,CPF)
TEXTTSS(CA-TOP SECRET SECURITY)
DUFPGM(*NONE*)
TNG MONITOR(OFF)
LIBRPROT(ON)
TSS9661I CA-Top Secret POWER      Status
PWRNODE(*NONE*)        NJEUSER(*NONE*)
JOBACID(U,7,0)          SUBACID(U,7)
TSS9661I CA-Top Secret PASSWORD  Status
NEWPW(MIN=4,WARN=03,MINDAYS=01,NR=0,ID,TS,RS)
HPBPW(000)              MSUSPEND(NO)
PWEXP(030)              PWHIST(03)
PWVIEW(YES)
TSS9661I CA-Top Secret FACMODE  Status
FACMONDE(B W,S F,T F,C F,J F,K F,UNF,APF,HSF,CNF)
TSS9661I CA-Top Secret CPF      Status
CPF(INACTIVE)
TSS0300I MODIFY  FUNCTION SUCCESSFUL
```

4.7 Using Control Options to Change Your Security Environment

The set of CA-Top Secret control options is the primary mechanism for defining the processing environment under which CA-Top Secret will operate. Through control options a security administrator can:

- Control how CA-Top Secret will process normally and how it will process under specific MODEs and circumstances;
- Indicate which features, facilities, and products are on the operating system;
- Control how individual facilities will be handled by CA-Top Secret;
- Specify password selection rules and violation thresholds;
- Set suspension thresholds for users
- Issue commands that force CA-Top Secret to reset after shutdown, or reinitialize after installing CA-Top Secret maintenance.

All control options have default values—that is, their respective values in the as-delivered version of CA-Top Secret. (The default values of the FACILITY control option vary by facility.) In general, these default values are consistent with a FAIL MODE security environment. It is recommended that certain control options be reviewed when CA-Top Secret is installed. The *Installation Guide* lists these control options. Verify that their values are appropriate to the specific conditions and requirements of your installation and to the initial phase of your security implementation.

CA-Top Secret protects security-sensitive control options against unauthorized entry and change. Only the MSCA or an authorized SCA has the automatic and absolute authority to select and change all control options.

There are three ways to select and change control options:

- through the CA-Top Secret Parameter File when CA-Top Secret is started
- from a VSE console through the TSS MODIFY command for the CA-Top Secret subsystem;
- through the TSS MODIFY command function.

Each method is explained in this section.

Note: Changes to control options made through console commands such as TSS MODIFY are temporary and end when CA-Top Secret is brought down. To effect a control option change that will span, for example, end of day shutdowns, add the entry to the Parameter File or the PARM field of the CA-Top Secret started task.

4.7.1 Restricted Control Options

Any control option that changes the security environment is restricted. Conversely, only control options that request CA-Top Secret status displays (for example, STATUS, VERSION) aren't restricted.

When a restricted control option is specified or changed at the VSE console, CA-Top Secret displays message TSS9080A, which prompts for the necessary authorization. The request will not take effect unless one of the following is entered:

- An ACID that possesses the CONSOLE attribute followed by that ACID's password in the format **ACID/password**.
- **Or**, the MSCAs **previous** password

Whenever a control option is changed, CA-Top Secret automatically maintains an audit trail, including the ACID that is changing the control option.

If a restricted control option is to be changed using the TSS MODIFY command function, then the administrator must possess the CONSOLE attribute.

Assigning the CONSOLE Attribute: Unless you are the MSCA, a security administrator needs MISC9(CONSOLE) authority to assign the CONSOLE attribute. Potential candidates for the CONSOLE attribute are the SCA and operations supervisors.

In the following example, the MSCA gives SCA01 MISC9(CONSOLE) authority to assign the CONSOLE attribute to those in his scope.

```
TSS ADMIN(SCA01) MISC9(CONSOLE)
```

4.7.2 Selecting Control Options Through the Parameter File

This optional method is intended to be used during start up or installation to specify the values of those control options that differ from their installation default values. It is especially useful when there are many control options whose default values are being changed. Instructions for its use are provided in the *Installation Guide*.

Any time that CA-Top Secret is started up it will reference the Parameter File and the PARM field of the started task procedure EXEC statement for overrides to the control option defaults. Changes to control options made through console commands and the TSS MODIFY command functions are effective only until CA-Top Secret is brought down. These changes aren't reflected when CA-Top Secret is subsequently restarted. In case of a conflict between the Parameter File and the PARM field, the PARM field prevails in determining the value of a control option.

For permanent changes you must code the changes in the Parameter File. Otherwise, use one of the following methods to changes control options temporarily:

- TSS MODIFY command from the VSE console
- TSS MODIFY command

The Parameter File should be owned and update access permitted to a limited number of ACIDs.

Note: Changes to the parameters in this file cannot be audited by CA-Top Secret.

If you are starting CA-Top Secret as a subtask, you must code SUB=MSTR on the startup command.

Changing Control Options Through the TSS MODIFY Command: This method of changing control options can be used at any time while CA-Top Secret is running. Moreover, it overrides control option values specified through the Parameter File, the PARM field, and the O/S START command.

The format for using the TSS Modify command is:

```
83 TSS,option,option,...
```

For example, the following command establishes WARN MODE as the global security environment, and tells CA-Top Secret to perform an automatic backup of the Security File at 5 a.m.

```
83 TSS,MODE(WARN),BACKUP(0500)
```

The two previous examples assume that TSS has an outstanding reply id of 83 on the VSE console.

Changing Control Options Through the TSS MODIFY Command: This method of changing control options can be used at any time while CA-Top Secret is running. It overrides control option values specified through **any other method**. Changes to control options made through TSS Modify entries are temporary and end when CA-Top Secret is brought down. TSS MODIFY can be entered like any other TSS command function in the format:

```
TSS MODIFY(option,option,...)
```

For example, the command below instructs CA-Top Secret to perform a daily backup of the Security File at 5 a.m.

```
TSS MODIFY(BACKUP(0500))
```

The MODIFY commands with the last two methods are dynamic and can happen at any time. Either method can be used to modify any parameter; therefore, either method can override the other.

Refer to the *Command Functions Guide* for detailed instructions on using TSS MODIFY.

Chapter 5. Password and System Entry Security

5.1 Password Security

The objective of a good security policy is to reduce the likelihood that your system will be compromised by unauthorized personnel. To realize this goal, it is important that users not give their ACIDs and passwords to other users (who may not have the same authorities as the other ACID) and to change their passwords frequently and at regular intervals.

CA-Top Secret requires password protection for all ACIDs by default. In addition, each password must have an expiration interval. An *expiration interval* is the number of days before CA-Top Secret forces a user to change his password. To let the user know his password is about to expire, CA-Top Secret displays a password expiration warning message on the job log or terminal. At this point, the user can change his password—on, before, or after the day the password expires. However, once a user's password does expire, he **must** provide a new password before he can sign on to CA-Top Secret.

A wide variety of password security policies can be supported by using the password protection controls and options provided by CA-Top Secret and discussed in this section. This section explains how to:

- Define and change passwords
- Set expiration intervals
- Maintain password history
- Use the Restricted Password List
- Set up and use masks
- Generate passwords randomly
- Select a password violation threshold

By combining these options to meet the special requirements of your site you can minimize the exposure or intentional disclosure of a user's password, thus protecting the integrity of your environment.

5.1.1 Defining Passwords

It is the responsibility of the security administrator to assign passwords to the ACIDs she defines. When defining ACIDs, the security administrator:

- Must assign a password to each user or Control ACID being defined.
- Can set an expiration interval of 1 to 255 days on a user-by-user basis, or let the interval default to the system value.
- Can specify a minimum number of days a user must wait after changing his password before the user can change it again.
- Can specify EXP so a user's password expires immediately when it is used, forcing a prompt to supply a new password.

Password options are specified using the PASSWORD parameter with either the CREATE or REPLACE function. (The PASSWORD parameter and its suboptions are discussed in detail in the *Command Functions Guide*.) Which function you use depends on whether the user is a new one or an existing one.

For a New User Use the CREATE command function. This function is covered in the section *How to Define a Password for a New User*.

For Existing Users Use the REPLACE command function. This function is covered in the section *Changing Passwords*.

5.1.1.1 How Password Creation Defaults Are Set

The guidelines that must be followed when users change their passwords, and the way in which CA-Top Secret responds to password changes, is determined by the defaults set using the NEWPW control option and its suboptions. If your site has chosen to use the default values for the NEWPW control option, the following rules for creating a password are automatically enforced:

- The password must be at least four characters long.
- The same letter can't be repeated in succession.
- Passwords can't match any of the entries in the Restricted Password List.
- Passwords can't be changed more often than once each day with these exceptions:
 - All security administrators can change their passwords as often as desired.
 - Users who select the random password feature can change their passwords multiple times within one day.

- Passwords can't match the userid or the first four characters of any word in the associated NAME field. For example, John Smith whose ACID is USER01 can't use either John, Smith, or USER01 as part of his password.
- Warning messages are issued beginning three days before the password's expiration.
- Passwords can't be too similar to previous passwords. For example, "mean" would be too similar to "lean".

You can change these default values and specify additional rules for CA-Top Secret to apply whenever a new password is selected by a user or a security administrator. The suboptions used to specify these default values and additional rules are briefly described in the section *Summary of Password Options*.

5.1.1.2 How to Define a Password for a New User

To define a password for a new user, use the PASSWORD parameter with the CREATE command function as shown below.

```
TSS CRE(USER01) NAME('GEORGE SMILEY') DEPT(PAYROLL)
      FAC(TSO) PASSWORD(TINKER, , EXP)
```

In this example, the password is TINKER; the missing second suboption tells CA-Top Secret to use your site's default expiration interval for all subsequent passwords the user selects; the EXP suboption indicates that TINKER will expire the first time it is used.

To change your site's default expiration level, use the PWEXP control option. How to specify PWEXP is covered in the *Control Options Guide*.

Setting an Expiration Interval: To use an expiration interval other than your site's default, specify a number from 1 to 255 for the second suboption as shown below.

```
TSS CRE(USER01) NAME('GEORGE SMILEY') DEPT(PAYROLL)
      FAC(TSO) PASSWORD(TINKER, 9, EXP)
```

In this example, the password TINKER will expire the first time USER01 signs on. However, by specifying 9 for the second suboption instead of allowing it to default, CA-Top Secret will force the user to enter a new password after nine days.

Note: The maximum number of days you can specify for the expiration interval is 255. For a password that never expires, you must specify 0 for the expiration interval.

If you don't want TINKER to expire immediately but still want the nine day expiration level, specify:

```
TSS CRE(USER01) NAME('GEORGE SMILEY') DEPT(PAYROLL)
      FAC(TSO) PASSWORD(TINKER,9)
```

Assigning a Password That Doesn't Expire: If the ACID you are creating needs a password that doesn't expire, specify 0 for the expiration interval as shown in the following example.

```
TSS CRE(USER01) NAME('CICSPRD1') DEPT(PAYROLL)
      FAC(BATCH,STC) PASSWORD(TINKER,0)
```

Creating an ACID Without a Password: If the ACID you are creating doesn't require a password (for example, it is a Region ACID or an ACID that automatically signs on a terminal), use the NOPW suboption for PASSWORD as shown below.

```
TSS CRE(USER01) NAME('TERMINAL01') DEPT(PAYROLL)
      FAC(TSO) PASSWORD(NOPW,0)
```

Whenever you specify the NOPW suboption, you must also specify 0 for the expiration interval or the password will expire using the default expiration interval set when CA-Top Secret was installed.

Note: It is not advisable to have many ACIDs with the NOPW suboption. In addition, ACIDs with this suboption should be restricted to ensure the ACID is used properly.

5.1.2 Changing Passwords

As stressed earlier, it is important to the integrity of your system for users to change their passwords frequently and at regular intervals. This section explains how a security administrator can change an ACID's password and the expiration interval that is associated with it. Other password options, such as maintaining the password history of an ACID and keeping users from changing their passwords are also covered in this section.

In addition to the default values set using the NEWPW control option, CA-Top Secret always enforces certain rules for users changing their passwords. These rules are:

- The new password can't be the same as the one you are changing.
- The new password can't match any of a number of previous passwords (see 5.1.2.2, "Maintaining Password History" on page 5-8 for details).

Whenever a password is changed (by you or the user), CA-Top Secret automatically changes the user's Security Record to reflect the new password.

There are numerous reasons why you may need to change a password for a user rather than letting the user do it himself. The most common reason is when a user forgets his password. If you need to change a password for a user, you must use the PASSWORD parameter with the REPLACE command function.

Note: You can also use the ADD command function to change a password, but using the REPLACE command function is the preferred method.

The following example shows how to immediately change the current password of USER01 to SPY.

```
TSS REP(USER01) PAS(SPY)
```

Since no expiration level is specified in this example, the assigned password will expire using the current default expiration interval set when CA-Top Secret was installed.

Setting an Expiration Interval: To set an expiration interval other than the default, specify a number from 0 to 255 for the second suboption as shown in the next example.

```
TSS REP(USER01) PAS(SPY,9,EXP)
```

In this example, the password SPY will expire the first time it is used. However, by specifying 9 for the second suboption instead of allowing it to default, the new password selected by the user will expire every nine days. As a result, USER01 will automatically be prompted to change his password immediately, and then again after each 9 day interval.

Note: The maximum number of days you can specify for the expiration interval is 255. For a password that never expires, you must specify 0 for the expiration interval.

If you don't want SPY to expire immediately but still want the nine day expiration level, specify:

```
TSS REP(USER01) PAS(SPY,9)
```

Changing a Password That Doesn't Expire: If the ACID whose password you are changing needs a password that doesn't expire (for example, it is a Region ACID or an ACID that automatically signs on a terminal), specify 0 for the expiration interval as shown below.

```
TSS REP(CICSPRD1) PAS(SPY,0)
```

When a Password Isn't Required: If the ACID does not require a password (for example, it could be an unsophisticated or a dumb terminal), use the NOPW suboption with PASSWORD as shown below.

```
TSS REP(USER01) PAS(NOPW,0)
```

Whenever you specify the NOPW suboption, you must also specify 0 for the expiration interval. If you don't, when your site's default expiration interval is reached, CA-Top Secret will prompt for a password the next time this ACID is used to sign on.

5.1.2.1 Reverifying a Changed Password

To further protect your security environment, you can require that users re-enter a password immediately after changing it. This process is called *password reverification*.

In the password reverification process, whenever a user changes her password CA-Top Secret verifies the change by immediately prompting her to re-enter it. If the password is entered incorrectly, the user is given a specified number of chances to enter it correctly. When the specified threshold is reached, the user is signed off of CA-Top Secret and must complete the entire logon sequence to enter the system.

To activate password reverification, use the NPWR suboption of the FACILITY control option. Set the threshold value using the NPWRTHRESH control option; the default is two attempts. For more information on these control options, refer to the *Control Options Guide*.

Note: Password reverification is available for CICS facilities only.

5.1.2.2 Maintaining Password History

For system integrity, it is important to maintain password history to prevent users from reusing previous passwords when their current password expires. *Password history* refers to how many previous passwords CA-Top Secret remembers for each ACID. For example, if you maintain a history of three passwords, when a user changes her password the new one must be different from the last three passwords she used.

To maintain password history use the PWHIST control option. You can specify from 1 to 64 previous passwords; the default is 3.

There are two ways you can change the setting of the PWHIST control option:

- dynamically from within CA-Top Secret, or
- directly in the Parameter File.

To change PWHIST dynamically, issue the command function shown below.

```
TSS MODIFY(PWHIST(2))
```

How to change the Parameter File is covered in Chapter 8, *Maintaining Special Security Records*.

5.1.2.3 Displaying a User's Password

Some sites give security administrators the ability to view the passwords of the users within their scope of authority. Provided strict security practices are followed, this feature is useful in certain situations (for example, when a user has forgotten the spelling of his password).

Note: The value for the PWVIEW control option must be YES before this option can be used.

If you are a security administrator who has been given this ability, you can view the passwords of the users within your scope by issuing the command function shown below.

```
TSS LIST(USER01) DATA(PWVIEW)
```

Note: When you display a password, CA-Top Secret alerts you that the password has never been used, or has been used (sometimes frequently) but never expired, by showing a password expiration date of 01/01/80 but a correct expiration interval.

5.1.2.4 Keeping Users From Changing Their Passwords

You can revoke the default—which allows users to change their own passwords—for selected users or for all users of the system. When a user isn't allowed to change his password, new passwords can be selected only by an authorized security administrator or by the automatic random password facility.

For Specific Users: To prohibit specific users from selecting new passwords, use the NOPWCHG parameter with either the CREATE or ADD command function. This parameter can be attached to either individual users or to profiles. The following example prevents an existing user, USER01, from changing his password.

```
TSS ADD(USER01) NOPWCHG
```

For All Users: To prohibit all users of the system from selecting new passwords, set the NU suboption of the NEWPW control option.

5.1.3 Specifying Optional Password Features

In addition to the default values automatically set for the NEWPW control option, you can specify additional password controls. These controls include:

- Listing words and prefixes that can't be used as passwords
- Defining password masks
- Generating passwords randomly
- Forcing passwords to be validated while using DORMANT or WARN mode
- Selecting a password violation threshold

This section contains instructions for using each of these options.

5.1.3.1 Changing the Restricted Password List

CA-Top Secret provides a Restricted Password List containing prefixes and words that can't be used as passwords. The function of the list is to prevent the use of obvious passwords such as company or facility names, months, and so on. CA-Top Secret won't allow a user to assign a new password that matches any of the prefixes or words on this list. For example, using the word NEW—a default entry on the list—you couldn't specify the word NEW or any word that begins with NEW (like NEWton) as a password. However, you could specify any word that contains the word NEW (like RENEW).

The Restricted Password List provided by CA-Top Secret has 33 default entries. These entries are listed below.

APPL	IBM	PASS
APR	JAN	ROS
ASDF	JUL	SEP
AUG	JUN	SIGN
BASIC	LOG	SYS
CADAM	MAR	TEST
DEC	MAY	TSO
DEMO	NET	VALID
FEB	NEW	VTAM
FOCUS	NOV	XXX
GAME	OCT	1234

You can specify up to 478 additional prefixes to be included in this list for a maximum number of 511 list entries.

How to Activate the Restricted Password List: To begin using the Restricted Password List, you must first specify the RS suboption for the NEWPW control option. You can then change the entries on the list either temporarily or permanently.

- To temporarily add or remove items from the list or to display the contents of the list, use the RPW control option. The Restricted Password List will remain changed until CA-Top Secret is recycled.
- To add or remove an entry on a permanent basis, you must modify the Parameter File. How to change the Parameter File is covered in Chapter 8, *Maintaining Special Security Records*.

To Add an Entry: To temporarily add one or more items to the Restricted Password List issue the TSS MODIFY command function with the RPW control option as shown below.

```
TSS MODIFY(RPW(ADD,PRES,CIA))
```

The sample command shown above adds the prefixes PRES and CIA to the list. As a result, PRES and CIA can't be specified as new passwords or used at the beginning of a new password. However, users currently using PRES or CIA as their password or the prefix of their password can still use it to sign on.

Note: Changing the Restricted Password List requires CONSOLE authority. If entered from an operator console, CA-Top Secret prompts you for your ACID and password to validate whether or not you have the authority to perform the changes. This step is omitted when TSS MODIFY is used because the validation was performed when you signed on.

If the list is full (you have reached the 133 entry maximum), CA-Top Secret displays a message to that effect. You must remove one or more current entries before you can add new entries to the list.

Tip Check for and remove any duplicate entries. For example, novocain, novice, November, and nova are all duplicates since they are all covered by the default prefix NOV.

To Remove an Entry: To temporarily remove one or more items from the Restricted Password List issue the MODIFY command function with the RPW control option as shown below.

```
TSS MODIFY(RPW(REMOVE,PRES))
```

The example shown above deletes PRES from the list. PRES can now be selected as a new password.

The validation step may be omitted in certain circumstances. See the note on 5-10.

To Display List Contents: To display the current contents of the Restricted Password List, enter the TSS MODIFY command function as shown below.

```
TSS MODIFY(RPW(LIST))
```

5.1.3.2 Using Masks to Define a Password

You can dictate the type of character that CA-Top Secret will accept for each position in a password by using the *mask* feature. A mask controls the placement of consonants, vowels, numbers, and special characters within a password. The password mask is enforced system-wide for all users except the MSCA, and applies to all new passwords—including those that are randomly generated.

How to Define a Mask: To define a mask, specify the MASK= suboption with the NEWPW control option. To construct the mask you must decide what type of character will be accepted for each position of a password. The character types you can use in a mask are listed on the next page.

Type	Stands For
a	any letter
c	consonant
v	vowel
n	number
x	non-vowel
?	any character

Note: Valid characters for non-vowels are consonants, numbers, and the special characters @, #, \$, |, {, and }. The special characters %, !, (, and) can't be used.

For example, **MASK=cvc??n**, tells CA-Top Secret that all new passwords must have:

- a consonant as the first character.
- a vowel as the second character.
- a consonant as the third character.
- any letter, number or special symbol for positions 4 to 6.
- a number for the last character.

The password MAT2B@3 would correspond to this mask.

You should design your mask so that passwords are somewhat recognizable and easier to remember, thus eliminating the need for a user to write her password down. The following example would allow such passwords as CATOX, ZULUK, and BESEY to be specified or randomly generated, but would not allow H2NO3 or 6#AK@. In addition, this example usually creates a password that is pronounceable.

NEWPW(MASK=CVCVC)

You should coordinate the composition of your mask with the NEWPW suboptions you have specified. For example, specifying **MASK=vnvn** could generate the password A5I6. In this case, the mask would override the NV and NM suboptions of the NEWPW control option if they were specified.

5.1.3.3 Using Random Password Generation

A CA-Top Secret feature lets new passwords be randomly generated by the system according to a specified password mask. There are two methods that can be used to randomly generate passwords:

- the user can request a randomly-generated password at any time, or
- a random password can be automatically generated the first time a user signs on after his current password expires.

Preparing for Random Generation: The ability to generate random passwords is controlled by the RNDPW suboption of the FACILITY control option. CA-Top Secret specifies RNDPW by default for the CICS facilities.

In addition to the RNDPW suboption, a password mask and minimum days must be specified before the random password generation option can be used. A password mask is specified using the MASK= suboption, and minimum days is specified using the MINDAYS= suboption of the NEWPW control option.

To automatically generate a password after the current one expires requires the RN suboption of the NEWPW control option to also be specified.

The following commands show how to specify the RNDPW, RN, MASK, and MINDAYS suboptions using the FACILITY and NEWPW control options.

```
FAC(fac=RNDPW)
NEWPW (RN,MASK=cvcv,MINDAYS=5)
```

Note: Random password generation is the only method available for changing a password if a user has the NOPWCHG attribute or if the NU suboption of the NEWPW control option is specified.

User-Requested Generation: The user can request that a new password be randomly generated at any time. Using this method, the user can change her password multiple times within one day (bypassing the once-a-day rule for changing passwords).

To randomly generate a new password, the user must:

1. Enter **random** as her new password. The following messages are displayed when ENTER is pressed:

```
TSS7030I Password Changed
TSS7020I Random Password About to be Displayed. Hit Enter to Continue.
```

2. Press the ENTER key again. The following messages are displayed:

```
TSS7021I Your New Password is VANA27
TSS7022I Memorize Password & Hit Enter Key - DO ***NOT*** RECORD
```

Memorize the generated password. Without this password the user can't sign on again.

3. Press the ENTER key. These messages are displayed:

```
TSS7000I acidname Last-Used mm/dd/yy hh:mm System=xxxx Facility=xxxxxxx  
TSS7001I Count=xxxxx Mode=xxxx Locktime=xxxxx Name=xxxxxxxxxxxxxxxxxxxxx
```

Automatic Password Generation: CA-Top Secret randomly generates a password for users the first time they sign on after their password has expired if NEWPW(RN) is specified. Even if this feature is active, the user can still manually change her password at any time.

To **only** allow passwords to be randomly generated (the user can't select her own password), all of the following items must be specified:

- The facility must have RNDPW specified.
- The user's ACID must have the NOPWCHG attribute.
- The NEWPW control option must have the RN suboption specified.

5.1.3.4 Forcing Password Validation By Mode

CA-Top Secret offers designated facilities the option of failing invalid passwords when operating in DORMANT or WARN mode. Using this option will keep a job from processing if a user omitted his password or supplied an incorrect one. It is especially recommended that you continue to use the defaults for this option to force password validation in all modes for all online facilities.

DORMANT Mode: To implement forced password validation in DORMANT mode, use the DORMPW suboption of the FACILITY control option. The following example shows how to turn this option on for the BATCH facility.

```
FACILITY(BATCH=DORMPW)
```

WARN Mode: To implement forced password validation in WARN mode, use the WARNPW suboption of the FACILITY control option. The following example shows how to turn this option on for the BATCH facility.

```
FACILITY(BATCH=WARNPW)
```

5.1.3.5 Selecting a Password Violation Threshold

The password violation threshold feature of CA-Top Secret secures entry to your environment by keeping "password guessers" out of your system. The *violation threshold* is the maximum number of consecutive times a wrong password can be entered while trying to sign on before the ACID is suspended.

How to Set a Threshold: To set a password violation threshold use the PTHRESH control option. You can set a threshold of one to 254 times; the default is three times. To change the default threshold, issue the following command:

```
TSS MODIFY(PTHRESH(2))
```

In this example, the default threshold of three was changed to two. As a result, the user can enter an incorrect password two consecutive times. If the user enters the correct password on his third try, CA-Top Secret allows him to sign on. However, if the user enters an incorrect password on his third try, CA-Top Secret suspends his ACID.

Note: The password violation threshold doesn't apply when attempting to assign a new password and the user fails to match the password selection rules.

Reinstating Suspended ACIDs: You can reinstate an ACID that has been suspended because the user exceeded the password violation threshold by performing the following two steps in any order.

1. Replace the user's current password by issuing:

```
TSS REPLACE(user-acid) PASSWORD(new-password | newpass,,EXP)
```

2. Remove the suspension by issuing:

```
TSS REMOVE(user-acid) SUSPEND
```

5.1.4 Summary of Password Options

This section summarizes and briefly describes the defaults and additional options you can specify for passwords using the NEWPW control option. A detailed description of each of these suboptions appears in the *Control Options Guide*.

The following NEWPW suboptions are set by default:

MIN=	Determines minimum password length. The default is four characters.
MINDAYS=	Determines the minimum number of days between password changes. The default is one day.
WARN=	Determines the number of days before a password expires that the user is notified of the impending expiration. The default is three days.
NR=	Indicates the number of repeating pairs allowed (for example, rabbit has one repeating pair—bb—while AABBCC has three). The default is 0, or no repeating characters.
RS	Prevents the use of passwords from the Restricted Password List.
ID	Specifies that the new password can't contain the ACID or parts of the name field.
TS	Prevents a user from specifying a password that is too similar to his previous password.

The following suboptions of NEWPW provide additional password controls.

NM	Specifies that only numbers can be used for passwords. Can't be specified with SW.
NO	Turns off a password mask.
NU	Specifies that users can't change their passwords.
NV	Specifies that vowels can't be used in the new password.
RN	Specifies that new passwords will be randomly generated by CA-Top Secret. MASK= must also be specified.
SW	Specifies that a new password must contain a national character (@,#,\$). Can't be specified with NM.
MASK	Dictates what types of characters are accepted for each position of a password. When defining a mask, consider these suboptions: NR=, NM, NV, and SW.

5.2 System Entry Security

The information in this section focuses on who can sign onto the system and from where the signon can be performed (called point of entry). The entire process is called *system entry validation*.

The first, and only required, method used to validate system entry is identifying who is trying to sign on. The user signing on is identified by entering the proper ACID/password combination—which CA-Top Secret validates as part of the signon procedure. If the user can't supply a combination that matches an entry in the Security File, entrance into the system is denied. However, if the user **can** supply a matching combination, CA-Top Secret allows the user to enter the system while adhering to all of the permissions and restrictions associated with that ACID.

The remaining methods used to validate system entry are optional and include:

- monitoring entry by day or time of use
- monitoring entry by device
- monitoring entry using job validation
- monitoring entry within the network

Which method, or combination of methods, you use to validate system entry should be dictated by your security policy. The rest of this section contains instructions for using each method.

5.2.1 Monitoring Entry Through Facility Authorization

CA-Top Secret considers a facility as a way of grouping options and associating them with a particular service that users sign on to. Examples of such services are BATCH and CICS. To sign on to a service, a user **must** be given access to the facility. Only the MSCA can access any facility by default. Everyone else must be explicitly authorized to access one or more facilities.

There are many options within the facility definition that can be used to tailor who has access to the facility and how security is to be handled for users of the service associated with that facility. See Chapter 11, *Protecting Facilities*, for information about these options.

How to Authorize Access: To assign authority to access a particular facility, use the FACILITY parameter with either the CREATE or ADD command function. The example shown below uses the ADD command function to authorize USER01 to access the BATCH facility.

```
TSS ADD(USER01) FAC(BATCH)
```

To give USER01 access to all facilities, issue this command:

```
TSS ADD(USER01) FAC(ALL)
```

For a complete list of facilities that can be specified with the FACILITY parameter, refer to the *Control Options Guide*. For a list of facilities that can be used for your site, check your Facility Matrix Table.

5.2.2 Monitoring Entry by Time of Use

By default, an authorized resource can be used at any time. However, you can control when an ACID can enter the system through a facility or device by specifying these time-related parameters:

DAYS	By day of the week
TIMES	By time of day
FOR/UNTIL	For a particular length of time
CALENDAR	For specifying many dates within a calendar year
TIMEREC	For specifying multiple time intervals within a 24-hour day

You can also specify additional options—like whether the ACID's actions are to be audited; see Chapter 3, *Violations, Logging, Reporting, and Auditing* for details.

5.2.2.1 By Day of Week

To limit an ACID's access to a facility by day of the week, use the DAYS parameter with the CREATE or ADD command function. The following example allows USER02 to access the CICS facility on Monday, Wednesday, and Friday.

```
TSS ADD(USER02) FAC(CICS) DAYS(MON,WED,FRI)
```

5.2.2.2 By Time of Day

You can further limit an ACID's access to a facility with the TIMES parameter—which specifies the time during the day that a facility can be used. The following example allows USER02 to access the CICS facility on Monday, Wednesday, and Friday, but limits access during those days from 9:00 a.m. to 5:00 p.m.

```
TSS ADD(USER02) FAC(CICS) DAYS(MON,WED,FRI) TIMES(09,17)
```

Note: Time restrictions are designated on a 24 hour clock; therefore, instead of specifying "9 to 5" you would actually specify "09,17".

5.2.2.3 For a Length of Time

If an ACID needs to access a facility or device for a certain length of time or until a particular date, use the FOR or UNTIL parameter. In the next series of examples, the first one allows the ACID to access CICS on Monday, Wednesday, and Friday from 9:00 a.m. to 5:00 p.m. for the next ten days.

```
TSS ADD(USER02) FAC(CICS) DAYS(MON,WED,FRI) TIMES(09,17)
FOR(10)
```

The second example gives the same access to CICS until the specified date, rather than for the next ten days.

```
TSS ADD(USER02) FAC(CICS) DAYS(MON,WED,FRI) TIMES(09,17)
UNTIL(12/31/99)
```

This example allows USER01 to use terminal PD01 until December 31, 1999.

```
TSS PER(USER01) TERM(PD01) UNTIL(12/31/99)
```

The following example further restricts use by allowing USER01 to use the terminal until the indicated date, but only between the hours of 9 a.m. and 6 p.m.

```
TSS PER(USER01) TERM(PD01) UNTIL(12/31/99) TIME(09,18)
```

5.2.2.4 To Correct Time Zone Discrepancies

Use the TZONE parameter to automatically correct a difference between the user's time zone and the CPU's time zone. For example, if USER01 is using a California terminal communicating with a New York CPU and should be restricted to using this terminal during normal business hours, use the TZONE parameter to correct the discrepancy between the time zones. The following two entries restrict USER01 to using terminal RD1 between 8 a.m. and 5 p.m. Pacific Coast Time.

```
TSS ADD(USER01) TZONE(-3)
TSS PER(USER01) TERMINAL(RD1) TIME(08,17)
```

Note: The TZONE parameter can only be used with a CREATE or ADD command function (not PERMIT).

5.2.2.5 For Calendars or Time Records

In addition to specifying particular days of the week or a single time interval, you can use the more specific CALENDAR or TIMEREC keywords. By using the CALENDAR keyword rather than the more general DAYS keyword, a PERMIT can be granted for specific dates of a year. In the same way, a TIMEREC allows the administrator to specify multiple 15 minute intervals in a 24-hour day when the PERMIT applies.

The CALENDAR or TIMEREC specified does not need to exist to do the permission; however the PERMIT will not execute correctly until the named record exists in the Static Data Table (SDT) and has been updated in storage using the TSS MODIFY(SDTTABLE) command.

Note: On a TSS PERMIT, the use of the DAYS and CALENDAR keywords, and the TIMES and TIMEREC keywords are mutually exclusive.

To replace existing permissions that use the DAYS or TIMES keywords with permissions that use the CALENDAR and TIMEREC keywords, you just need to revoke the old permissions and re-issue the resource permission using the new keywords. Sample commands are shown below.

```
TSS PERMIT(USR01) DSN('INV.MASTER.FILE') ACCESS(UPDATE)
                  CALENDAR(CAL1)
```

```
TSS PERMIT(USR01) DSN(SFT.) ACCESS(ALL) TIMEREC(TIME1)
```

CALENDAR and TIMEREC records are created using the TSS ADD(SDT) command with the appropriate keywords. These records are then stored in the Static Data Table (SDT). For detailed instructions on defining calendars and time records, see Section 8.3, Static Data Table (SDT) Record.

5.2.3 Monitoring Entry by Device

This section explains how to restrict signons to specific devices, how to control multiple signons by the same ACID, and how to protect terminals and readers from unauthorized access. CA-Top Secret lets you protect terminals or readers from unauthorized access in several ways: automatic locking, manual locking, and special user identification devices (all explained in this section), and resource ownership (explained in Chapter 7, *Protecting Resources*).

5.2.3.1 How to Restrict Signons to Specific Devices

CA-Top Secret can be used to control system entry by restricting access to terminals, readers, CPUs, and nodes. Access can be restricted for these device types:

- CPUs
- Online terminals (VTAM, BTAM, VM local, logical, and bi-synch)
- Remote (RJE) and local JES readers
- Internal readers
- NJE nodes

The specific resource class name (CPU, TERMINAL, VMRDR, and NODES) is used as the keyword for determining access authorizations and restrictions for that resource.

How to Limit CPU Access: The CPU parameter is used to establish access authorizations for CPUs. Authorization to access the CPU can be qualified with specific limitations (such as through a particular facility, on certain days, or during certain times). The following is an example of designating a batch-only machine accessible to the day shift:

```
TSS PERMIT(DAYPROF) CPU(SYSA) FAC(BATCH) TIME(08,17)
```

How to Limit Terminal Access: You can protect terminal devices using two independent methods: the TERMINAL parameter and the SOURCE parameter. Both methods are explained here.

The TERMINAL parameter is used to establish access authorizations for terminals and readers. Once defined to CA-Top Secret, access to terminals can be specifically restricted. The following example authorizes USER01 to use New York terminals in the morning.

```
TSS PER(USER01) TERM(NYC) TIMES(07,12)
```

To permit all users connected to the PAYROLL profile (PAYPROF1) access to a local terminal (address K61L1234) in the Personnel Office from 7:00 a.m. to 11:00 a.m., enter:

```
TSS PERMIT(PAYPROF1) TERM(K61L1234) TIMES(07,11)
```

Source of Origin: Designated jobs and online users (or profiles) can be restricted to enter the system from designated terminals and readers. This option is called *source of origin security* and is implemented using the SOURCE parameter with the ADD command function. The following example restricts USER01 to signing on only from a terminal whose ID is C5NY002.

```
TSS ADD(USER01) SOURCE(C5NY002)
```

Note: Adding SOURCE to a profile will affect all users attached to that profile.

Terminal C5NY002 doesn't have to be defined to CA-Top Secret to be a source. However, if the specified terminal is a protected resource, then be sure the user is permitted to access it:

```
TSS PER(USER01) TERM(C5NY002)
```

Ownership is explained in Chapter 6, *Resource Security Validation*.

A source of origin restriction overrides TERMINAL authorization. In other words, if USER01 has a source restriction of C5NY002 and USER01 is permitted to access terminal C5NY007 as shown above, an attempt by USER01 to sign on from C5NY007 would result in a security violation.

How to Limit Node Access: The NODE parameter is used to establish access authorizations for nodes.

5.2.3.2 How to Control Multiple Signons

With CA-Top Secret, you can prevent a user from having concurrent signons to the same multi-user online region within a facility. This feature keeps a user from signing on to the same region from multiple terminals.

To prevent concurrent signons, use the SIGN(S) suboption of the FACILITY control option. The following example prevents concurrent signons to the CICSTEST facility.

```
TSS MODIFY(FAC(CICSTEST=SIGN(S))
```

5.2.3.3 Default ACID Signon

The DEFACID feature of CA-Top Secret allows a default ACID to be used for a signon when the userid entered has not been defined. This feature provides a powerful implementation tool which allows a signon to occur, although all users are not explicitly defined.

It should also be used with some care to avoid its misuse. DEFACID is controlled at the facility level through the FACILITY suboption DEFACID. For more details, refer to the *Control Options Guide*.

If a signon is entered and the userid is not defined, CA-Top Secret will check to determine if a facility DEFACID has been specified. If it is defined, then CA-Top Secret will attempt to complete the signon using the DEFACID. All normal restrictions that apply to the signon will be enforced, for example, facility, date, time and source.

In addition, if the DEFACID is assigned a password, and password checking is in effect, then the DEFACID password must be supplied by the signon.

5.2.3.4 Automatic Terminal Signon

The Automatic Terminal Signon (ATS) procedure is used for terminals from which an explicit signon is neither possible nor desirable. The ATS procedure is invoked whenever a protected transaction is entered from such a terminal.

When the user attempts to use a protected transaction, and has not signed on, the Automatic Terminal Signon process performs the following:

- Searches the Security File for an ACID which exactly matches the terminal ID which the user is attempting to access.

If found, CA-Top Secret performs the following checks:

- Facility
- Terminal (if owned)
- Source (if ACID source is specified)
- Duplicate signon security (when facility has set the sub-option SIGN(S))
- Suspended or expired attributes
- If access is allowed for all checks, signon is performed without issuing the "last used" or "status messages" from CA-Top Secret.
- If an automatic terminal signon ACID is not found in the Security File, the facility DEFACID can be used under certain circumstances. DEFACID used in the context of Automatic Terminal Signon requires no password checking. Consult the *Implementation: DLI Guide*, the *Implementation: CICS Guide*, and the

Implementation: Other Interfaces Guide, for specific details on whether and how the DEFACID can be used for Automatic Terminal Signon with DL/1, CICS, and CA-IDMS, respectively.

5.2.3.5 How to Lock and Unlock Terminals

Unattended terminals can be protected against unauthorized access by using the CA-Top Secret terminal locking option. Terminal locking prevents use of the terminal until it is either logged off or unlocked.

Terminal locking can be triggered automatically by CA-Top Secret or manually by command. CA-Top Secret provides several TSS command functions and a control option that allow the security administrator and individual users to control when inactive or unattended terminals are locked.

For security administrators:

- Use the LTIME parameter with the ADD command function to set terminal lock times for individual users.
- Use the LOCKTIME suboption of the FACILITY control option to set lock times for all terminals connected to a specific facility.
- For CICS only, use the LTLOGOFF suboption to sign off the terminal if the LOCKTIME expires a second time before the user enters his password.

For individual users:

- The TSS LOCK/UNLOCK command functions let users lock and unlock their terminals anytime.

The standard locktime procedure is:

1. When you sign on a terminal, CA-Top Secret begins to monitor LOCKTIME thresholds.
2. When the LOCKTIME threshold is expired, at the next action key (ENTER, CLEAR, PF key, and so on) the terminal screen is cleared and you are prompted for your password.
3. In CICS, if the LTLOGOFF suboption of the FACILITY control option is set to YES, the terminal is signed off security and logged off if the LOCKTIME expires again.

5.2.3.6 Automatic Locking

CA-Top Secret can automatically lock a terminal that has been inactive for a pre-established duration. Locking is available for most online facilities (including VM, TSO, CICS, DL/1, and CA-Roscoe).

Automatic locking thresholds can be established at both the user and facility level. Different inactivity locking thresholds can be designated for each facility. Individual user locking thresholds can also be set, which override the facility level threshold values. How to set facility and user locking thresholds are explained below.

Facility Locking Thresholds: Facility locking thresholds are designated in minutes using the LOCKTIME suboption of the FACILITY control option. Terminal activity durations are based upon the time elapsed between two commands or transactions. (The viewing or editing of data doesn't constitute a command.)

The following example specifies that terminals logged on to TSO lock if CA-Top Secret doesn't detect activity after five minutes.

```
FAC(TSO=LOCKTIME=5)
```

User Locking Thresholds: User—or profile—locking thresholds are designated by using the LTIME parameter with the ADD command function. **The LTIME parameter overrides any applicable facility level setting** (which was set via LOCKTIME). Values can range from 0 to 120 minutes. The following example instructs CA-Top Secret to lock any terminal that has been inactive for 15 minutes, and that is logged on to be a user attached to PROF01.

```
TSS ADD(PROF01) LTIME(15)
```

Once the 15 minutes expires, the user must supply her password at the CA-Top Secret prompt to re-enter the system.

Locking Terminals by Facility: CA-Top Secret also supports LTIME by facility, causing a user's signed on terminal to lock, if unused, after a specified time with a given **facility**. The facility is identified by the second suboption of the LTIME keyword. The example below locks the signed on user's terminal after 15 minutes of inactivity on the CICSQA facility **only**.

```
TSS ADD(USER03) LTIME(15,CICSQA)
```

To Override Locking: You can override automatic terminal locking for specific ACIDs by setting LTIME to 0 for that ACID. The following example cancels automatic terminal locking for USER03 when logged on to TSO.

```
TSS ADD(USER03) LTIME(0,TSO)
```

5.2.3.7 Manual Locking

To prevent unauthorized access to the terminal while it is unattended, a user can lock her own terminal by entering the LOCK command function. Any attempt to use a locked terminal results in immediate termination of the command or transaction before it executes.

To lock a terminal enter:

```
TSS LOCK
```

Note: A locked terminal can be signed off.

5.2.3.8 Unlocking a Locked Terminal

To unlock an automatically or manually locked terminal, issue the TSS UNLOCK command function from the locked terminal. CA-Top Secret will prompt you to enter your current password.

5.2.3.9 How to Use Special Authentication Devices

CA-Top Secret can support security devices that require user voice or image identification. You must customize CA-Top Secret using the Application Interface or the installation exit to implement these supplements.

Operator ID Cards: CA-Top Secret supports the physical identification of users through operator identification cards. This feature can be used to supplement password security and is available for VM and CICS through IBM 3270-compatible terminals. It is implemented using the OIDCARD parameter.

The command shown below causes CA-Top Secret to prompt for the operator ID card to be inserted into the terminal's badge reader whenever USER01 signs on.

```
TSS ADD(USER01) OIDCARD
```

5.2.4 Monitoring Entry by Job Validation

Batch Job Validation: A batch job must be associated with an ACID so that CA-Top Secret can tell which facilities and resources it can access and how they can be accessed. To CA-Top Secret, a batch job's ACID is simply another ACID with an associated Security Record, and a set of specific access authorizations. All the system entry restriction options can be specified for a User ACID—including facility, source of origin, and CPU restrictions.

Online Job Submission Validation: For jobs submitted through CICS or through another online facility that uses the POWER internal reader (even batch jobs or started tasks submitting other batch jobs), CA-Top Secret provides an additional "layer" of security control beyond the basic batch job validation. The focus of this security layer is whether the submitter has the authority to submit the job. In other words, CA-Top Secret checks whether the ACID of the submitter is authorized to submit using the ACID associated with the job. If he isn't, the job will be flushed from the system before it is initiated.

Submitting a Job for Another User: By default, a defined user is only allowed to submit jobs for execution under his own ACID. Explicit authority is required to allow a user to execute jobs using other ACIDs—called *surrogate processing*. This authority is granted using the ACID parameter and the PERMIT command function. The example shown below gives WYLMIO1 the authority to submit jobs using ACID SMITH01.

```
TSS PERMIT(WYLMIO1) ACID(SMITH01)
```

Program pathing is an access authorization option used to restrict the performance of certain functions—in this case job submission and scheduling—to a specific program called a *privileged* program. Surrogate processing can use program pathing restrictions to provide additional security for job submission and scheduling activity by using the PRIVPGM parameter with the ACID parameter on the PERMIT command function. In the following example, PROD01 is given the authority to submit jobs using ACID PAY10M, but only through the program SCH007P (a job scheduling program).

```
TSS PER(PROD01) ACID(PAY10M) PRIVPGM(SCH007P)
```

For a complete explanation of program pathing, see Chapter 7, *Protecting Resources*.

5.2.5 Monitoring Entry Within the Network

If your site uses an NJE network, you must ensure that all work entering or leaving each node complies with your site's security policy. This policy determines how inbound and outbound work, as well as jobs that are stored and forwarded, are protected.

This section focuses on how entry to other network nodes is secured. There are two primary methods you can use:

- the Secured Signon feature for generating substitute passwords, and
- node processing for controlling incoming NJE jobs and SYSLST.

5.2.5.1 Defining Users to Multiple Nodes

Since only users defined to the target node can initiate jobs on that node, there are several ways CA-Top Secret lets you define users to multiple nodes:

- If CA-Top Secret isn't running on all nodes, you must explicitly define the user to all nodes the user needs to access. This method entails much administrative work, since anytime the user's security information changes, it must be updated manually on each system.
- If CA-Top Secret is running on all nodes and each node has its own Security File, you can use the Command Propagation Facility (CPF) to propagate security information for a user across all nodes while only entering it once. CPF is discussed in detail in Chapter 10, *Using the Command Propagation Facility*.
- If CA-Top Secret is running on all nodes which share the same Security File, then each user is implicitly defined to every other node in the system.

5.2.5.2 Using the Secured Signon Feature

CA-Top Secret supports the Secured Signon feature of the IBM Network Security Program (NETSP) that must use the NETSP keyword. This feature lets network users from one or more nodes communicate with a VSE host using a *PassTicket*. A *PassTicket* is a dynamically generated, one-time-only, password substitute with a limited lifespan. Using *PassTickets* eliminates the need to send an ACID's password across the network when signing on to other nodes, and allows CA-Top Secret to validate the user by checking the generated *PassTicket*. *PassTickets* can be used to access any facility where a password is used to sign on.

How to Set Up the Environment: All systems using PassTickets must have identical application names and session keys for all nodes on the network.

In addition, all systems must use the same standard rules to establish the validity of the PassTicket. Check your IBM documentation if you are unfamiliar with these rules.

Generating a PassTicket: To generate a PassTicket, the PassTicket generator algorithm must be used. This algorithm requires specific information as input data: the user's ACID, the time of day, and a session key. This produces a PassTicket that is used as a substitute for a specific end-user password.

There are two ways to generate a PassTicket using this algorithm:

1. If you are running on an MVS system, you can use the RCVTPTGN callable service provided by CA-Top Secret to generate the PassTicket on the host.
2. You can create a program that incorporates the algorithm for any function that generates a PassTicket. This method allows the PassTicket to be generated on a network.

How to Define PassTickets: To define PassTickets to CA-Top Secret you must identify each application that can accept a PassTicket and assign that application a unique password called a *session key*. This information is added to the Node Descriptor Table (NDT) using the PSTKAPPL (PassTicket application) and SESSKEY (session key) parameters of the TSS ADD command function. Both parameters must be supplied. Refer to the *Command Functions Guide* for detailed information on these keywords.

In the following example, a PassTicket application for TSO consists of the literal 'TSO' and a four-character SMFID. The SMFID, in this particular case, is known as SYSA; therefore, the PassTicket for that system is TSOSYSA. The session key is 296LFD.

```
TSS ADD(NDT) PSTKAPPL(TSOSYSA) SESSKEY(296LFD)
```

PSTKAPPL Defines the application id. Depending on the application, the secured signon function uses a specific method to determine the application id:
For CICS or APPC applications, the application id is defined using the standard naming conventions you use to define these applications in a VTAM APPL statement.

For VSE batch jobs that include TSS passwords in the JCL, you can replace the password with a PassTicket. The application id for batch jobs is defined by prefacing the adapter CPU id of the system with the characters VSE. For example, VSEA is the application id for all batch jobs on a machine with adapter CPU id A.

SESSKEY Defines the encryption key for the application. It is an eight-byte key and **must** be specified in the format of 16 hexadecimal digits.

The Node Descriptor Table (NDT) contains all PassTicket application and session key-related node information. The NDT is a global record similar to the Resource Descriptor and Field Definition Tables. See Chapter 1, *Introducing CA-Top Secret* for an explanation of the NDT.

Chapter 6. Resource Security Validation

Once you have defined users to CA-Top Secret and assigned system entry restrictions to them, the next step in building your security database is to secure your resources. In CA-Top Secret securing resources is a two step process:

Step 1 Define the resources to CA-Top Secret by verifying that the resource class is defined to the Resource Descriptor Table (RDT) and assigning ownership of that resource to a particular ACID.

Step 2 Permit access authorizations depending on who needs to access each resource (and, in some cases, how, when, and from where they can access it).

Access authorizations to a resource are implemented using the TSS PERMIT command function and customized through the associated parameters.

Depending on the resource class, you can restrict a user's access to that resource by day, time, facility, program, or access level. Resource protection can also be extended on a default or global basis.

After a resource has been defined and the appropriate authorizations issued, any future requests to access that resource are filtered through the Security Validation Algorithm. Access depends on what PERMITs the ACID has, how explicit those PERMITs are, and where they are stored (in the user, profile, or ALL Record). The Security Validation Algorithm uses all of this information to determine the "best fit" between what resources the user is allowed to access and what resources the user is requesting to access.

This chapter explains:

- How to assign, remove, and transfer resource ownership.
- What access restriction options you can use to tailor your security environment.
- What the Security Validation Algorithm is and how it works.
- How "best fit" is determined using the Security Validation Algorithm.

Defining Resources: The process of identifying a resource to CA-Top Secret is called *defining* and is a crucial step in securing your environment, since CA-Top Secret won't protect a resource it doesn't know about. However, you can specify blanket protection for all resources in a resource class through the DEFPROT or RESCLASS parameter.

Unlike the one step process of defining users to CA-Top Secret through their ACIDs, defining resources is a two step process.

Step 1 The resource class is defined to the Resource Descriptor Table (RDT) (if it isn't a predefined resource class).

Step 2 Ownership of each resource in the resource class is then assigned to an ACID.

Resource Descriptor Table (RDT): By default, the CA-Top Secret Resource Descriptor Table (RDT) already contains a number of predefined resource classes. Each of those resource classes is identified by a unique keyword, and has certain attributes associated with it. 6.1.5, “List of Predefined Resources” on page 6-16 lists most of the predefined protected resource classes and corresponding resource class keywords for MVS, VM, and VSE.

You can extend default protection to all resources predefined in the RDT (whether the governing mode is WARN, IMPLEMENT, or FAIL) by assigning the DEFPROT attribute to the specific resource class. This is called *default protection of resources* and means that each resource in that class will have security protection even if it isn't defined to CA-Top Secret—that is, a security violation will occur if a request is made to access the resource. (In FAIL MODE default protection exists only for data sets and volumes.) The DEFPROT attribute is explained in the section *Providing Default Protection* later in this chapter.

Note: Several CA-Top Secret interfaces for CA-Unicenter products are also defined in the RDT and are reserved for CA product use. For a complete list of these resources, refer to the CA-Top Secret *Command Functions Guide* and the *security administrator's Guide* for the respective CA products.

Customizing the RDT: Using the TSS command, you can customize the RDT by modifying existing resource attributes or adding new resource classes; how to customize the RDT is covered in Chapter 8, *Maintaining Special Security Records*.

How to View RDT Contents: To view the contents of your RDT, issue the TSS LIST(RDT) command. To display information about a specific resource class, issue:

```
TSS LIST(RDT) RESCLASS(resource-class-name)
```


6.1 Owing Resources

Most resources must first be owned before their use can be authorized. (The principal exceptions are commands, which are protected through the Limited Command Facility and USERx class resources.) Either the TSS CREATE or ADD command functions must be used to assign ownership. Use CREATE when an ACID is being defined; use ADD anytime.

Note: The ACID to which the resource is being added (or removed) must be within the scope of the administrator.

Topics discussed in this section include:

- Assigning ownership
- Locating a resource's owner
- Transferring ownership
- Removing ownership

6.1.1 Assigning Ownership

Once you know that the resource class is defined to the RDT, the next step is to determine which ACIDs should have ownership of the individual resources within that class. Assigning ownership lets you individualize access restrictions for particular resources within the resource class.

All of the ACID types discussed in Chapter 4 can own a resource; however, how you distribute resource ownership throughout the ACID hierarchy is very important. The next few sections discuss:

- How to select resource owners.
- How to use generic prefixing and masking to minimize data entry.

6.1.1.1 Selecting Resource Owners

Zone, Division, Department, Profile, and User ACIDs can **own** resources. Of these ACID types, however, only users and profiles can be given authorization (PERMITted) to **access** resources (Zone, Division, and Department ACIDs can't sign on). Therefore, resources should typically be owned by appropriate Department, Division, or Zone ACIDs, and PERMITted to User and Profile ACIDs on an as-needed basis. Following this approach facilitates simpler and more effective security administration.

Tip: Allow users to own all files matching their unique high-level qualifier.

By default, ownership of a resource automatically gives full (ALL) access of that resource to the owner. An exception to the default is when a masked resource is owned at a user level. In this situation, the owner doesn't have access to the resource and can't be permitted access to it. Therefore, it is recommended that all masked resources be owned at a department, or higher, level.

If ownership is granted to a User ACID, that user automatically has complete access to the resources. For example, if USER01 was given ownership of the PAYROLL.UPDTE data set with the command shown below, then USER01 has ALL access to the PAYROLL.UPDTE data set.

```
TSS ADD(USER01) DSN(PAYROLL.UPDTE)
```

In addition, this access can't be overridden with these commands:

```
TSS PERMIT ..... ACTION(DENY)
TSS PERMIT ..... ACCESS(NONE)
```

Note: Since ownership by a profile would imply total access to the resource for every user attached to that profile, it is recommended that profiles should never own anything.

Ownership of a resource by a Department ACID, does **not** imply automatic access to that resource for all users in that Department, because a Department ACID (like Zone and Division ACIDs) represents a group of ACIDs and isn't an actual user. A Department ACID doesn't sign on or function as an individual user does; therefore, it can't access the resource. Resources are assigned to a Department ACID much the same way books are placed on a bookshelf—for safekeeping.

When Department ACIDs own resources, each user in that department has to be explicitly authorized to access that resource through the PERMIT command function. This enables you to restrict access to the resource on an as-needed basis.

To illustrate this concept, the following example designates the Financial Department (FINDEPT) as the owner of the INVEST.RES data set. If USER02 belongs to that department he would still have to be authorized (via the TSS PERMIT command function) to access the INVEST.RES data set. Therefore, USER02 can be given UPDATE access or even restricted to READ only access to this data set. The same concept applies to resource ownership by Division and Zone ACIDs.

```
TSS ADD(FINDEPT) DSN(INVEST.RES)
```

Further discussion of the PERMIT command function and access level restrictions can be found in 6.3.2, “Restricting Access” on page 6-28.

Tip: If a particular department is to be given ownership of many resources that will be PERMITted many times (over 500 times), it is a good idea to create several dummy departments and split up the ownership of these resources among them. This enhances processing efficiency by achieving a more balanced distribution of information on the CA-Top Secret Security File.

6.1.1.2 Using Generic Prefixing

Assigning ownership for each individual resource may sound like an enormous task, but by using generic prefixing, masking, and Profile ACIDs whenever possible you can greatly reduce the number of TSS command entries required. This section explains how to use generic prefixing. Profile ACIDs are covered in Chapter 4, *Setting Up and Modifying Your Security Environment*, and masking is covered in 6.1.1.3, “Using Masking” on page 6-9.

Resources can be defined to CA-Top Secret either by their full name or through a generic prefix. A generic prefix is a high-order substring of the full resource name. If your site has implemented and enforced sound resource naming conventions, then you can use generic prefixing extensively to specify resources. Generic prefixing allows a group of resources with similar names in the same resource class to be defined to CA-Top Secret simultaneously. (Instead of specifying each individual resource, you can identify a group of resources with one ADD statement.) For example, the IEHPROGM, IEHINIT, and IEHLIST programs in the PROGRAM resource class (abbreviated below as PGM), can be grouped under the generic prefix IEH. That way, instead of entering:

```
TSS ADD(SYSDEPT) PROG(IEHPROGM)
TSS ADD(SYSDEPT) PROG(IEHINIT)
TSS ADD(SYSDEPT) PROG(IEHLIST)
```

you would only have to type:

```
TSS ADD(SYSDEPT) PROG(IEH)
```

You can then permit access to all of the resources starting with IEH by entering:

```
TSS PER(USER02) PROG(IEH)
```

USER02 is now permitted to access programs IEHPROGM, IEHINIT, IEHLIST, and so on.

How to Specify a Generic Prefix: Generic prefixing can be used with any resource class—including commands and transactions. CA-Top Secret allows identical generic prefixes if they are used with different resource types (for example, PROG(PAYROLL) and APPL(PAYROLL) don't conflict because the resource type is different).

The minimum length permitted for a generic prefix is one character—except for data sets and volumes where the minimum length is two characters. For most resource classes the maximum prefix length is eight characters; an exception is data sets, where the maximum prefix length is 26 characters.

The following examples show how generic prefixing is used with programs and data sets.

Program Example: The following example assigns ownership of any program that begins with the prefix IEH (like IEHLIST and IEHPROGM) to the ACID DEPT01.

```
TSS ADD(DEPT01) PROG(IEH)
```

Data Set Examples: The first data set example shows how all data sets used by the Publications Department could be prefixed by TECHPUBS.

```
TSS ADD(PUBDEPT) DSN(TEHPUBS)
```

Once defined, you can PERMIT a user to access any data set prefixed by TECHPUBS, as shown below.

```
TSS PERMIT(USER01) DSN(TEHPUBS) ACCESS(ALL)
```

Using this method eliminates the need for separate PERMIT commands to access both of these data sets: 'TEHPUBS.PROD.SCEDS' and 'TEHPUBS.GRAPHICS.SCEDS'. For further refinement, a prefix can span several data set name index levels. For example, instead of specifying DSN(TEHPUBS), you could specify DSN(TEHPUBS.PR) to reduce the number of data sets a user can access.

Undercutting: *Undercutting* refers to the use of a generic prefix used to establish ownership that is generically higher (more inclusive) than an existing prefix (for example, the prefix IMS is more inclusive than the prefix IMSTEST). If the undercut is valid, CA-Top Secret automatically transfers ownership of the specified resources to the new owner. Therefore, be careful not to inadvertently undercut a prefix that is already owned (for example, if IMSTEST is already owned, assigning ownership of IMS undercuts the ownership of IMSTEST).

When using undercutting, you must adhere to these rules:

- All prefixes must be owned within the scope of the administrator specifying the undercut.
- You can't use a generic prefix to establish ownership that is generically lower (less inclusive) than an existing prefix (for example, the prefix IMSTEST can't be used to undercut the prefix IMS).

Note: Undercutting restrictions don't apply when the subsequent prefix is used to specify resource authorization rather than resource ownership. For example, the following entries are valid:

```
TSS ADD(DEPT01) DSN(IMS)
TSS PER(USER02) DSN(IMSTEST)
```

How to use undercutting to transfer ownership is discussed in the section *Transferring Ownership of Resources* later in this chapter.

The GENERIC and NONGENERIC Attributes: The GENERIC and NONGENERIC attributes are used to indicate whether or not a resource class supports generic prefixing—GENERIC indicates that prefixing is supported, while NONGENERIC deactivates prefixing. These attributes only affect permissions and don't affect the way resource ownership is designated or how the CICS Bypass List is processed.

Note: Only the GENERIC attribute can be changed through the TSS REPLACE(RDT) command function. See Chapter 8, *Maintaining Special Security Records*, for more details.

To illustrate how these attributes work, you can permit a user to resource PROG(PSRV), which would allow access to PSRV, as well as PSRVTEST or any other PROG whose first four characters match the prefix PSRV. However, if the NONGENERIC attribute is activated, a permit to PROG(PSRV) **only** allows the user to access PSRV and not PSRVTEST.

When changing the resource class from GENERIC to NONGENERIC, or from NONGENERIC to GENERIC, the previous security permissions are retained for all existing definitions. In addition, previous permissions will list differently to reflect that they contain the GENERIC or NONGENERIC attribute.

Specifying the NONGENERIC Attribute: To have a general resource treated as a fully qualified name rather than as a generic prefix, use the NONGENERIC attribute. The NONGENERIC attribute supports both long and short resource classes. Refer to *Appendix B: Prefixed Resources*, for a list of resources that can't be used with the nongeneric attribute.

The NONGENERIC attribute applies to the following resources **by default**:

IUCV
VMCF
VMDIAL
VMMACH
VMRDR

To add the NONGENERIC attribute to a resource class, enter:

```
TSS REPLACE(RDT) RESCLASS(PROGRAM) ATTR(NONGENERIC)
```

In this example, the NONGENERIC attribute is assigned to the PROGRAM resource class. By using the NONGENERIC attribute, a permit to PROG(PSRV) would allow access **only** to PSRV.

Overriding the NONGENERIC Attribute: If the PROGRAM resource class in the RDT has the NONGENERIC attribute and you wanted to permit programs IEHPROGM, IEHINIT, and IEHLIST to one ACID, enter:

```
TSS ADD(USER01) PROG(IEH(G))
```

The G indicates that IEH is a generic prefix and not a fully qualified resource name.

Removing the NONGENERIC Attribute: To remove the NONGENERIC attribute, use the REPLACE(RDT) command function. The following example removes the NONGENERIC attribute by replacing it with the GENERIC attribute.

```
TSS REPLACE(RDT) RESCLASS(PROGRAM) ATTR(GENERIC)
```

6.1.1.3 Using Masking

For frequently protected and numerous resources (like data sets), CA-Top Secret provides an additional way to minimize TSS command entries—masking. Unlike generic prefixing—which can be employed with any type of resource—masking (or patterning) can only be used for certain types of resources. These resources are listed in *Appendix B: Prefixed Resources*.

Tip: To see whether or not a resource supports masking, use TSS LIST(RDT) to check the RDT for the MASK or NOMASK attribute. The default is NOMASK.

Masking is used to group sets of resources whose names share similar characteristics. However, masking isn't restricted to prefixes. The similar characters can occur in the beginning, middle, or end of the resource name, and there may be any number of variable characters in between. Special characters are specified to represent variations between resource names.

A masked resource name is treated by CA-Top Secret like a generic prefix. Any data set that **begins** with a pattern indicated by a mask is considered a match for it by the security validation algorithm, and the associated access authorizations are honored.

CA-Top Secret provides five different masking techniques:

- floating pattern
- variable character substitution
- index substitution
- fixed position
- ACID substitution

If you decide to use masking, you should make sure that you tell users which masks (or combination of masks) are being used, and how those masks work. How to use each technique is explained next.

Floating Pattern Masking: A floating pattern mask uses the special character "-" to represent a variable number of characters (including no characters). A floating pattern mask can also cross node (also called qualifier or index) boundaries. A floating pattern mask is the **only** masking technique that can't be combined with other masking techniques.

In the following example, the first entry indicates that a matching prefix must begin with the characters ACCT, followed by a variable number of characters represented by a "-", and must end with the characters VEND. The second entry indicates that a matching prefix must begin with the characters SAL, followed by a variable number of characters represented by a "-", and must end with the characters XMPT.

Entry:	Matches:	But not:
ACCT-VEND	ACCTPAY.VENDOR ACCTVEND	ACC.VEND AP.ACC.VEND (Doesn't begin with ACCT)
SAL-XMPT	SALPAY.XMPTOR.MST SAL.XMPT.MSTR SALXMPT.DEC.MST (Variable number of characters includes 0) SAL.MSTR.XMPT.QTR	SA.XMPT PR.SAL.XMPT (Doesn't begin with SAL) SAL.HOLE.ND

Variable Character Substitution: Variable character substitution uses the special character "*" to represent from zero to eight characters. Multiple asterisks can be strung in sequence to represent up to 44 characters. For example, *** could represent 0 to 24 characters. However, the left-most masking character can't be the second character (for example, A*M100 won't work), although the left-most masking character can be the first or third character.

Note: The count of the characters represented includes the periods that separate node levels.

In the following example, the asterisk in the first entry is used to represent zero to eight characters between ACCT and M; thus the entry matches ACCT.PAYM but not ACCTPAYD. In the second entry, *LAB doesn't match NEW.JERSEY.LAB since NEW.JERSEY. is more than eight characters long; you would need to specify **LAB to

match NEW.JERSEY.LAB. The third entry shows what would and would not match SAL*M.MARCH.

Entry:	Matches:	But not:
ACCT*M.DATA	ACCTPAYM.DATA	ACCTPAYD.DATA
*LAB	LAB	NEW.JERSEY.LAB
SAL*M.MARCH.	SALPAYM.MARCH SALM.MARCH SAL.C.M.MARCH	SALPAYD.MARCH SAL.MARCH SAL.EXEMPTS.J.MARCH (nine characters between SAL and M)

Note: Specified by itself, ***** (for example, PERMIT DSN(*****)) matches every resource in that class in the installation, since CA-Top Secret ignores the characters that follow the span of the mask in determining matches.

Index Substitution: The index mask is an extension of the variable mask and allows you to specify where index levels must appear in a resource name. Index substitution masking uses "*" to tell CA-Top Secret to ignore an index (node) level. The masking characters can be strung together to represent multiple index levels, for example, "*.*.". The characters "*.*" appearing at the beginning of the resource name are used to represent a one to eight character index; each asterisk appearing within the name can represent from zero to eight characters.

The first example specifies that .BALL must be prefixed by one to eight characters; thus, the data set BALL.GAME doesn't match this definition. The second example specifies that two indexes must appear between CICS. and .F. CICS.FEATURES contains no indexes between CICS. and .F, therefore it doesn't match the mask. The last example shows what would and would not match SAL.*.BKUP.

Entry:	Matches:	But not:
*.BALL	BASKET.BALL	BALL.GAME
CICS.*.*.F	CICS.RUM.TSS.FIL	CICS.FEATURES
SAL.*.BKUP	SAL.PAY.BKUP.ABC SAL.XMPT.BKUP SAL.US.REC.BKUP	SAL.BKUP SAL.BKUP.PAY SAL.BKUPLIB.A.BKUP (Nine characters between SAL and BKUP)

Fixed Position Substitution: Fixed position masking uses the special character "+" to represent single positions within a resource name. The following examples show what would and would not match A123+.TSO or SAL+++BKUP.

Entry:	Matches:	But not:
A123+.TSO	A1234.TSO	A123.TSO
SAL+++BKUP	SALPAY.BKUP SALREC.BKUP	SALRECV.BKUP SALRE.BKUP

ACID Substitution: ACID substitution uses the special character "%" to represent all or part of the ACID indicated in a TSS command. One use for this technique is to give users access to any data set whose prefix matches their ACID. For example, the TSS command:

```
TSS PER(USER01) DSN(ACCT.%.LOAD)
```

gives USER01 authorization to access data sets such as ACCT.USER01.LOAD and ACCT.USER01.LOAD.ABD, but doesn't give USER01 access to ACCT.USER02.LOAD.

For Entire ACID: The following procedure shows how to authorize full access to resources prefixed by their ACIDs for all TSO users at an installation.

Step 1 Assign the MSCA ownership of the "." character; ownership of "." won't allow the MSCA access to a particular resource unless that access is explicitly permitted. (Only the MSCA can own a special character.) Ownership for the data set resource is assigned to the MSCA by entering:

```
TSS ADD(MSCA acid) DSN(%.)
```

Step 2 PERMIT all TSO users to have all access to any resource prefixed by an ACID. The command shown below allows all TSO users access to any data set prefixed by their ACID.

```
TSS PER(ALL) DSN(%.) FAC(TSO) ACC(ALL)
```

For Partial ACID: CA-Top Secret recognizes a special technique to specify a portion of the userid to be used as a mask. The special technique uses the % character with values identifying the start and length of the userid being entered. For example, to permit TUSER01 access to data set USER01, enter:

```
TSS PER(TUSER01) DSN(%26%)
```

Combining Masks: Variable character, fixed position, index, and ACID substitution can be combined in any form. However, they can't be combined in any way with a floating mask. The following table shows various examples of how masking characteristics can be combined.

Entry:	Matches:	But not:
MAR++.*.SUM	MAR86.A.SUM MAR85.PERS.SUMM MAR86.WELFARE.SUMMER	MARCH.SUM MAR86.PERS.WELFARE.SUMM

Note: To permit data sets that are masked with either asterisks, plus signs, or percent signs in the initial position(s) of the data set, the MSCA must first be assigned ownership of the special prefix characters or their combinations. For example:

TSS ADD(msca acid) DSN(++*.)

would allow the entry of:

TSS PER(USER01) DSN(++*.)

which allows USER01 access to any data set with at least a two character prefix (and at least two qualifier levels).

The CA-Top Secret security validation algorithm only considers permissions that begin with one of the above special characters if it hasn't already encountered a data set match. Otherwise, it will ignore these types of entries.

6.1.1.4 How to Assign Ownership

Resource ownership implies that the owning ACID has an access level of ALL. (Access levels are discussed in 6.3.2.3, "By Access Levels" on page 6-30.) Since it may not be desirable to grant unlimited access to individual users or profiles, you should assign resource ownership to a Department or Division ACID using the ADD command function. Then, full or restricted resource access can be authorized for other, non-owning ACIDs using the PERMIT command function.

Assigning resource ownership can have either of the following results:

If Resource is:	ADD Command Function will:
New (undefined)	Define the resource by assigning ownership to the ACID specified in the command function.
Owned by another ACID	Transfer ownership to the ACID specified in the command function, and automatically PERMIT the previous owner to have full access to the resource.

How to assign ownership of individual resources is explained next.

How to Assign CPU Ownership: Once owned, a CPU can't be accessed unless explicit authorization is granted. The CPU parameter is how CPU ownership and authorizations are specified. CPUs should be identified by the AUDITID of their system.

Note: The AUDITID of a CPU consists of four characters. The first three are the constant **VSE**, the last character is a letter from A to Z or a number from 1 to 9 that corresponds to the System Adapter CPUID. For more information on setting up and maintaining this CPUID, please see your CA-CIS for VSE documentation.

The following example assigns ownership to DEPT01 of the CPU resource SYSA.

```
TSS ADD(DEPT01) CPU(SYSA)
```

How to Assign Terminal Ownership: The **TERMINAL** parameter is used to establish ownership of a terminal.

```
TSS ADD(DEPT01) TERMINAL(K18L3064)
```

How to Assign Volume Ownership: The **VOLUME** parameter is used to establish ownership of a volume.

```
TSS ADD(DEPT01) VOL(WORK01)
```

How to Assign Data Set Ownership: The **DSNAME** parameter is used to establish ownership of a data set.

```
TSS ADD(DEPT01) DSNAME(PROD.SMP.CNTL)
```

How to Assign Job Ownership: To give an ACID, USER01, ownership of jobs submitted from ALPHA2.USERJ*, enter:

```
TSS ADD(USER01) NODES(ALPHA.USERJ*)
```

6.1.2 Locating the Owner of a Resource

The owner of a resource can be found by using the TSS WHOOWNS command function. All resources that generically match the designated resource—and are within the scope of the administrator who issued the WHOOWNS—are displayed in alphabetical order.

All owned resources of a particular resource type within the scope of the administrator issuing the command can be displayed by using an * as the operand of WHOOWNS. For example, to determine all the owned programs within your scope enter:

```
TSS WHOOWNS PROGRAM(*)
```

6.1.3 Transferring Resource Ownership

You can transfer the ownership of a resource from one ACID to another by using the ADD command function with the UNDERCUT parameter. The UNDERCUT keyword is required whenever ownership is being transferred because it indicates to CA-Top Secret that the security administrator making the entry is aware that the resource is already owned. Since CA-Top Secret automatically transfers ownership of the specified resources from the former owner to the new owner, only the new owner needs to be identified in the command; no identification of the current owner is required.

Note: Both the new and former owners must fall within the scope of the administrator.

To illustrate this procedure, assume that terminal TSON0001 is currently owned by USER01, and ownership is to be transferred to USER02. Ownership is transferred by issuing the command shown below.

```
TSS ADD(USER02) TERMINAL(TSON0001) UNDERCUT
```

Whenever ownership is transferred, CA-Top Secret automatically gives full access authorization to the former owner (provided the former owner was a User or Profile ACID, and not a Zone, Division, or Department ACID).

Undercutting: You can also use undercutting to transfer the ownership of a set of resources. For example, if USER01 owns all IMSTEST programs, entering the commands shown below results in USER02 owning all IMS programs, including all IMSTEST programs. (Thus, ownership of the IMSTEST programs has been transferred by being undercut.)

```
TSS ADD(USER02) PROG(IMS) UNDERCUT
```

Note: If you forget to specify UNDERCUT, CA-Top Secret issues a message reminding you that it is needed.

When undercutting is used to transfer ownership, a PERMIT to the former owner is implied. Using the previous example, the following command is implied.

```
TSS PER(USER01) PROG(IMSTEST)
```

For details on the undercutting process, see 6-7.

How to Remove Access From the Former Owner: Use the NOPERMIT parameter if the former owner shouldn't be granted full access. In the following example, the former owner of the terminal TSON0001 retains no access authorization to it.

```
TSS ADD(USER02) TERM(TSON0001) UNDERCUT NOPERMIT
```

6.1.4 Removing Ownership

You can remove ownership by using the REMOVE command function. The following example removes job ownership from USER01.

```
TSS REM(USER01) NODES(ALPHA.USERJ)
```

6.1.5 List of Predefined Resources

The following sections list the predefined resources provided in the RDT for the MVS, VM, and VSE environments.

6.1.5.1 For VSE and/or MVS

The following table lists the predefined resource classes that apply to VSE, MVS, or both systems.

Table 6-1 (Page 1 of 3). Predefined MVS Resource Classes	
Type	Resource Class Keyword
APPC transaction programs and profiles	APPCTP
APPC side information files	APPCSI
APPC logical units	APPCPORT
APPC logical units	APPCLU
PC ports	CAPCLU
Data set	DATASET
PDS member names	PDSMEMn (n=1 to 5)
Tape or DASD volume	VOL
JES nodes or readers	TERMINAL and JESINPUT
TCAM, VTAM, or BTAM terminal	TERMINAL
Program	PROGRAM
CICS destinations	DCT
CICS files	FCT
CICS journal entries	JCT
CICS programs	PPT
CICS temporary storage areas	TST
IMS Application Group Name (AGN)	APPL
IMS Data Base Definition (DBD)	DBD
IMS Program Status Block (PSB)	PSB
IMS 4.1 commands	CIMS
CPU	CPU
Owned user-defined resource type	UR1, UR2
Field	FIELD
CA-Tape Scratch tape (CATAPE)	ABSTRACT
Linkage editor authorization (AC1)	ABSTRACT
Installation-defined system resource	ABSTRACT

Table 6-1 (Page 2 of 3). Predefined MVS Resource Classes	
Type	Resource Class Keyword
CA-1 security bypass control (XTD98000)	ABSTRACT
DB2 resources	DB2
CA-IDMS sub-schema	SUBSCHEM
CA-IDMS area	AREA
Operator commands	OPCMD
SMS managed volumes, database tokens	IBMFAC
DB2 resources	IBMGROUP and any keyword starting with DB2.
SMS management class	MGMTCLAS
VTAM applications	VTAMAPPL
Automatic ACID propagation	PROPCNTL
CICS CEMT functions	SPI
SMS storage class	STORCLAS
TSO logon account	TSOACCT
TSO user attributes	TSOAUTH
TSO performance group	TSOPRFG
TSO logon procedure	TSOPROC
SPF panel	PGM or CMD/XCMD
TSO commands	PGM or CMD/XCMD
CA-Roscoe commands (monitors)	PGM or CMD/XCMD
CICS transactions	OTRAN or TRAN/XTRAN
CA-IDMS transactions	OTRAN or TRAN/XTRAN
IMS transactions	OTRAN or TRAN/XTRAN
Unowned user-defined resource type	USERx (x=any keyboard character)
Nodes	NODES
CA product commands and programs	CACMD
Databases	DATABASE
System device allocation of graphics, teleprocessing, and unit record devices	DEVICES
Job submission and cancellation	JESJOBS

Table 6-1 (Page 3 of 3). Predefined MVS Resource Classes	
Type	Resource Class Keyword
JES spool data sets	JESSPOOL
ISPF, CICS, REXX panels	PANELS
Job submission	SURROGAT
User suppression of output labeling	PSFMPL
SDSF 1.3 commands, functions, and other resources	SDSF
Message transmission from one TSO user to another	SMESSAGE
IBM consoles	SYSCONS
Resource classes used by other CA product interfaces	USRCLASS
Job output monitoring, routing, and processing to local devices, RJE stations, or NJE nodes	WRITER
Data Lookaside Facility	DLFCLASS
PC access	CAPCID
Authorities	DB2SYS
Databases	DB2DBASE
Tables/Views	DB2TABLE
Plans	DB2PLAN
Packages	DB2PKG
Collections	DB2COLL
Buffer Pools	DB2BUFFP
Storage Groups	DB2STOGP
Tablespaces	DB2TABSP

6.1.5.2 For VSE

The following table lists the predefined resource classes for VSE.

Table 6-2 (Page 1 of 2). Predefined VSE Resource Classes	
Type	Resource Class Keyword
VSE Library	VSELIB

Table 6-2 (Page 2 of 2). Predefined VSE Resource Classes	
Type	Resource Class Keyword
VSE Sublibrary	VSESLIB
VSE Library Member	VSEMEMBR
VSE Partition	VSEPART

6.1.5.3 For VM

The following table lists the predefined resource classes for VM.

Table 6-3. Predefined VM Resource Classes	
Type	Resource Class Keyword
CP Commands	CPCMD
Virtual Machine Usage	VMMACH
Minidisks	VMMDISK
Discontiguous Saved Segments	DCSS
Diagnose Codes	DIAGNOSE
Dialed Virtual Machines	VMDIAL
DASD Volumes	VOLUME
RSCS Nodes	VMNODE
OS/DOS Data Sets	DATASET
VM Readers	VMRDR
CPUs	CPU
Terminals	TERM
IUCV	IUCV
VMCF	VMCF
Directory	DIRECTRY
SFS command	SFSCMD
Dataspace	DSPACE

6.2 Controlling Access

After identifying your resources to CA-Top Secret, the next step in constructing your security database is to assign access authorizations to each ACID that needs to use those resources. An ACID's authorization to access a resource is determined by the PERMITs found in his user record, in the profile records attached to his ACID, and the ALL Record (which lists globally accessible resources).

Using the PERMIT Command Function: The PERMIT command function is used to designate which resources an ACID is allowed to access. To enter the command you need to supply the ACID being granted access, the resource class, and the name of the individual resource. A sample command is shown below where the ACID is USER01, the resource class is TERMINAL, and the individual resource name is PD000001.

```
TSS PER(USER01) TERM(PD000001)
```

Note: You can also use generic prefixing when issuing PERMITs; for example, you could use the following command to permit USER02 to access all programs beginning with the IEH prefix:

```
TSS PER(USER02) PGM(IEH)
```

By adding the appropriate keywords to the PERMIT command function, you can restrict what resources the ACID can access and how they can be accessed. How to restrict access is covered in 6.3.2, "Restricting Access" on page 6-28.

The rest of this section explains:

- How to provide default protection.
- How to authorize access to a resource.
- How to restrict access to a resource.
- How to deny access to a resource.

6.2.1 Providing Default Protection

In FAIL mode, volumes and data sets are automatically protected by default. However, default protection isn't limited to volumes and data sets. To give default protection to predefined and dynamically defined resources that are normally not protected by default in any mode, including FAIL, assign the DEFPROT attribute to any resource in the RDT. For example, to provide default protection for data sets in IMPLEMENT mode, enter:

```
TSS REP(RDT) RESCLASS(DATASET) ATTR(DEFPROT)
```

This command generates access violations for those data sets that haven't yet been defined.

Note: When the DEFPROT attribute is added to a resource that is in WARN mode, access to the resource is still allowed but a warning message is generated.

An administrator can give default protection to any of the following predefined resources by modifying the appropriate entry in the RDT to include the DEFPROT attribute.

```
ABSTRACT
ACID
APPCPORT
APPCSI
APPCTP
APPLICATION
AREA
CACMD
CAPCID
CAPCLU
CIMS
CPCMD
CPU
DATABASE
DBD
DB2
DB2BUFFP
DB2COLL
DB2DBASE
DB2PKG
DB2PLAN
DB2STOGP
```

DB2SYS
DB2TABLE
DB2TABSP
DCSS
DCT
DEVICES
DIAGNOSE
DIRECTRY
DLFCLASS
DSNAME
DSPACE
FCT
FIELD
IBMFAC
IBMGROUP
IUCV
JCT
JESJOBS
JESSPOOL
MGMTCLAS
OPCMD
OPERCMD5
OTRAN
PANEL
PDSMEM1
PDSMEM2
PDSMEM3
PDSMEM4
PDSMEM5
PPT
PROGRAM
PROPCNTL
PSB
PSFMPL
RECIPIID
SDSF
SFSCMD
SMESSAGE
SPI
STORCLAS
SUBSCHEM
SYSCONS
TERMINAL
TSOACCT
TSOAUTH
TSOPRFG
TSOPROC
TST
UR1

UR2
 USRCLASS
 VMCF
 VMDIAL
 VMMACH
 VMMDISK
 VMNODE
 VMRDR
 VOLUME
 VSEPART
 VSELIB
 VSESLIB
 VSEMEMBR
 VSEUSER
 VTAMAPPL
 VXDEVICE
 VXFILE
 WRITER

Detailed instructions on modifying the RDT appear in Chapter 8, *Maintaining Special Security Records*.

6.2.2 Authorizing Access

Use the PERMIT command function to give users access to defined resources in either an unlimited or specifically restricted manner. Restrictions are specified by incorporating the appropriate parameters into the PERMIT command. For example, the following command allows USER01 read-only access to data set MAG.NET.FIELD through the TSO facility. Access is allowed only from 8 a.m. through 6 p.m. for a period of thirty days starting from the date this command was issued.

```

TSS PER(USER01) DSN('MAG.NET.FIELD') FAC(BATCH)
      ACC(READ) TIME(08,18) FOR(30)
  
```

Note: USER01 would require the ability to access BATCH through a previous CREATE/ADD FAC(BATCH) command for this example to be valid. If no facility restriction had been specified above, then USER01 would have been able to access MAG.NET.FIELD through any facility he can normally access.

You can also give an ACID the ability to administer resources that don't fall within her scope by specifying the ACTION(ADMIN) parameter on the PERMIT command function. Refer to Chapter 4, *Setting Up and Modifying Your Security Environment*, for a more detailed discussion of this parameter. For a list of all the parameters that can be associated with the PERMIT function refer to the *Command Functions Guide* or your *CA-Top Secret Reference Summary*.

6.2.2.1 Making Resources Globally Accessible

You may want to make certain types of resources (such as system and compiler libraries, and storage and work DASD volumes used for data set creation) available to all users. In CA-Top Secret, this is easily done by adding that resource to the ALL Record. All the standard access restrictions (for example, ACCESS, FACILITY, PRIVPGM, TIME, and so on) can be specified for globally accessible resources.

Note: Global authorizations don't actually update everyone's security records. Rather, they are maintained in common system memory (SVA), as well as on the ALL Record in the CA-Top Secret Security File. A change made to the ALL Record is effective immediately on the CPU from which the administrator makes the change. On other connected CPUs, the ALL Record isn't actually updated in SVA until a request is made to the Security File (for example, when a job or session initiates). Then, all of the connected CPUs are automatically synchronized as the new user or job initiates, or when any TSS command is executed.

The following table contains a list of resources that are commonly made globally accessible with the access restrictions they are usually assigned.

Resource	Access Restrictions
AUDIT data sets	ACCESS(READ)
procedure libraries	ACC(READ)
compiler libraries	ACC(READ)
production control libraries	ACC(READ) PRIVPGM(PGMA,PGMB)
system catalog data set	ACC(UPDATE)

6.2.2.2 Adding Resources to the ALL Record

Resources are added to the ALL Record by the PERMIT(ALL) command function. When you add a resource to the ALL Record, you can still specify other access restrictions, such as TIME and FACILITY, that would normally be available for that resource. When adding resources to the ALL Record, keep in mind that the resource being defined must be owned and that the PERMIT(ALL) command function **applies to both defined and undefined users**.

The following examples show how to add various resources to the ALL Record to make them globally accessible. Detailed instructions on maintaining the ALL Record appear in Chapter 8, *Maintaining Special Security Records*.

Data Set Example: This example makes the SYS1.BROADCAST data set available to all users from the BATCH facility at the UPDATE access level.

```
TSS PER(ALL) DSN('SYS1.BROADCAST') ACC(UPDATE) FAC(BATCH)
```

Volume Example: Storage and work volumes can be made globally accessible in this manner:

```
TSS PER(ALL) VOL(volume-list) ACC(CREATE)
```

Terminal Example: The next example allows **all** users—both undefined and defined—to access terminals with the PD01 and PD02 prefixes.

```
TSS PERMIT(ALL) TERM(PD01,PD02)
```

6.2.2.3 Overriding Global Authorizations

A specific access authorization in a User or Profile Record overrides a global authorization in the ALL Record. For example, the specific access authorization shown below for OPER01:

```
TSS PER(OPER01) DSN('SYS1.BROADCAST') ACC(ALL)
```

would override the following global authorization from the ALL Record (which provides only default READ access):

```
TSS PER(ALL) DSN('SYS1.BROADCAST')
```


6.3 Selective Revoke Command

You can issue a specific revoke to remove one permission. This is accomplished by listing the user or profiles XAUTH data, and then issuing the revoke **exactly** as it is listed.

For example, the output of the following TSS LIST command:

```
XA DATASET = SYS1.PROCLIB
ACCESS    = READ
PRIVPGM   = IEBCOPY
```

would enable the following authorization to be revoked:

```
TSS REVOKE(user) DSN(SYS1.PROCLIB) ACCESS(READ) PRIVPGM(IEBCOPY)
```

Even if there were other permissions with different access levels or other criteria, only this permission would be revoked.

6.3.1 Using Multiple Access Authorizations

Multiple authorizations can be used to efficiently "tailor" elements in an overall security structure. For example, you might have a large group of users that should be allowed to read a family of data sets, while only a few users in this group need the authority to update a particular data set in this family. Be aware, however, that multiple authorizations should be used carefully to avoid unexpected results.

For each resource access request, CA-Top Secret conducts a security validation search to determine whether the access request should be granted. It is during this search that multiple access authorizations are located. To eliminate authorization ambiguities while still allowing users great flexibility in setting up their security restrictions, CA-Top Secret uses a validation algorithm to determine whether a particular access request should be granted. Refer to 6.4, "The Security Validation Algorithm" on page 6-36 for a description of how this algorithm works.

When there are multiple access authorizations, CA-Top Secret treats separate PERMIT entries designating the same ACID and resource as discrete permissions rather than consolidating them into one combined permission. For example, these two sets of PERMITs are significantly different in terms of what authorizations are being established:

```
TSS PER(ALL) TERM(NYC) TIMES(08,17) DAYS(WEEKDAYS)

versus

TSS PER(ALL) TERM(NYC) TIMES(08,17)
TSS PER(ALL) TERM(NYC) DAYS(WEEKDAYS)
```

The first set is more restrictive than the second, since it uses **and** logic that requires both conditions to be satisfied (that is, to use the NYC terminals, you must log on Monday through Friday during business hours only). The second set uses **or** logic, which allows logon either during the week or during business hours.

Note that in certain instances, the historical order in which multiple authorizations were made can determine which CA-Top Secret error message will be issued. For example, assume these two PERMITs have been entered in the order indicated:

```
TSS PER(USER01) DSN(ABC) ACC(READ) DAYS(MON)
TSS PER(USER01) DSN(ABC) ACC(CRE) DAYS(TUES)
```

Now, if an attempt is made to create an ABC data set on a Monday, the CA-Top Secret error message issued indicates that the request was denied because USER01 did not have CREATE access. However, if the order of the authorizations had been reversed, then the error message would indicate that USER01 only had authorization to create ABC data sets on Tuesdays.

Revoking Multiple Authorizations: Multiple authorizations—that is PERMITs that specify both the same ACID and resource identifier—can be revoked by using a single TSS REVOKE function. For example, TSS REV(USER01) DSN(ABC) would revoke both of these PERMITs:

```
TSS PER(USER01) DSN(ABC) ACC(READ) DAYS(MON)
TSS PER(USER01) DSN(ABC) ACC(CRE) DAYS(TUES)
```

6.3.2 Restricting Access

By adding the appropriate keywords to the PERMIT command function, you can restrict that ACID's access by designating:

- Source of access (like a designated terminal or CPU).
- Time/Date of access (9 to 5, weekdays, until 5-31-99).
- Facility access.
- Path of access (through a particular program, loaded from a particular library).
- Access level (READ, WRITE, UPDATE, ALL, and so on).

- ACTION taken by CA-Top Secret when a particular access request is received.

Note: These restriction options are not available for all resource classes.

The options source, time, and date were discussed in Chapter 5, *Password and System Entry Security*; this section focuses on path of access, access level, and using the ACTION keyword. Facility was briefly discussed in Chapter 5 and is discussed further in this chapter. For a more in-depth discussion of access restriction options and their implementation, refer to your *Command Functions* and the facility-specific *Implementation Guides*.

6.3.2.1 By Facility

There are many options within the facility definition that can be used to tailor who has access to the facility and how security is to be handled for users of the service associated with that facility. To restrict a user from having access to a specific resource only through designated facilities, use the PERMIT or ADD command function with the FAC parameter. The specific resource class name—such as VMRDR, TERMINAL, or CPU—is used as the FAC operand for determining access restrictions to the specified facility. The resource can be indicated by its full identifier or a generic prefix.

Note: Facility suboptions override global control options of the same name.

Examples: To restrict an ACID to using only those terminals whose prefix is PD01 or PD02, add TERM(PD01,PD02) to the TSS PERMIT command function.

The next example shows how to restrict USER01 to accessing program SCH007P through the BATCH facility.

```
TSS PER(USER01) PROG(SCH007P) FAC(BATCH,STC)
```

Remember that USER01 must be allowed to have access to this facility through an entry such as:

```
TSS ADD(USER01) FAC(BATCH,STC)
```

6.3.2.2 By Program Path

The program pathing access authorization option allows access to certain types of resources or the performance of certain functions only through specific programs, called *privileged* programs. If a program other than a "privileged" one is used, the request causes a security violation.

The program pathing option is used most often to control access to data sets and volumes. However, other resources can also use this option. Program pathing could be used to

ensure, for example, that a CICS resource must be accessed only through a specific application program.

This option can also be used to provide additional security for job submission and scheduling activity. Access is restricted through the PERMIT command function and the ACID and PRIVPGM parameters. The following example allows PROD01 to submit jobs that will run under the ACID PAY10M, but only through program SCH007P, a job scheduling program.

```
TSS PER(PROD01) ACID(PAY10M) PRIVPGM(SCH007P)
```

Note: For program pathing to function, the issuer of the FRACHECK macro must pass the PRIVPGM name. Also, the program must be in the linklist if LIB= isn't specified.

The next example specifies the additional restriction that this ACID can only be submitted through CICS.

```
TSS PER(PROD01) ACID(PAY10M) PRIVPGM(SCH007P) FAC(CICS)
```

Note: The second example requires that PROD01 be allowed to access the CICS facility through a CREATE/ADD FAC(CICS) entry, as well as be authorized to access SCH007P through a TSS PER PGM(SCH007P) entry.

By using the PRIVPGM keyword with the LIBRARY and FACILITY keywords, you can design an ACID's access authorization so that ACID can only access a resource through a specific program, that must be loaded from a specific library, which is only accessed through a specific facility. In this example, USER02 must load the PAYUPDAT program from the PAYROLL.PRODLIB library, through BATCH to access the PAYROLL.MASTER data set.

```
TSS PER(USER02) DSN(PAYROLL.MASTER) PRIVPGM(PAYUPDAT)
LIB('PAYROLL.PRODLIB') FAC(BATCH) ACC(UPDATE)
```

6.3.2.3 By Access Levels

The types or levels of access to many resource classes—including data sets and volumes, CICS, CA-IDMS, and DL/1 resources, and user-defined resources (UR1 and UR2)—can be closely controlled through CA-Top Secret. The types of access levels to which you can restrict a user depend on the type of resource class involved, but some of the more common access levels are:

- READ (the default for data sets)
- WRITE

- UPDATE
- CREATE
- NOCREATE
- SCRATCH
- ALL
- NONE

Access levels are specified using the **ACCESS** parameter of the **PERMIT** command function. The following example uses the **ACCESS** parameter to authorize **USER01** to update—which also implies read—all data sets with **AP** as their highest-level qualifier.

```
TSS PER(USER01) DSN(AP.) ACC(UPDATE)
```

Note: Ownership automatically confers total access to the resource upon the owner unless Record Level Protection (RLP) has been implemented.

Additional Access Levels: For resources such as CICS and IMS databases, CA-Top Secret recognizes access level keywords that correspond to the actual system terminology (such as **BROWSE**, **REPLACE**, **PURGE**, and **FEOV**). In addition, CA-Top Secret also uses the three access level determinations described below.

CONTROL For VSAM data sets, **CONTROL** allows control-interval processing.

For **VOLUME** resources, when **CONTROL** and **CREATE** are both present, the combination allows the creation of data sets regardless of the applicable data set authorizations.

FETCH Enables access to a program library (load library) for the purposes of executing programs.

The **FETCH**-only access level option enables the owner of a program library to allow others access to it for the execution of programs—while preventing any other type of access. Thus, **FETCH**-only protection prevents the authorized user from reading the data set or copying a program for his own use (and, perhaps, modifying it to bypass special checking). Programs from **FETCH**-protected libraries can only be executed or invoked through other standard system invocations. All **FETCH** requests in **VSE** are translated as **READ** requests.

BLP Allows an OS/390 user to access a tape, bypassing the information kept in the label of that physical tape. **READ** and **UPDATE** are the only valid access levels for **BLP**.

For detailed information on access levels for all resources predefined to the RDT, refer to the *Command Functions Guide*.

Controlling Access at the Record and Field Level: Using the Record Level Protection (RLP) capabilities of CA-Top Secret, you can control which records within a data set—and even which fields within a record—a user has access to. RLP is implemented

by defining the records and fields you want to protect to the Static Data Table (SDT). Access is then controlled through the TSS PERMIT command with the appropriate access level specified. For detailed instructions on implementing RLP, see Section 7.6, Protecting Records and Fields. For information on the SDT, see Section 8.3, Static Data Table (SDT) Record.

Defining Unique Access Levels for Resources That are Dynamically

Defined: The CA-Top Secret-defined default values for access levels aren't the same as those passed by the ATTR= keyword of RACHECK. ATTR= values for RACHECK or FRACHECK are shown in the first two columns of the following table; their associated values for the TSS command are contained in the last two columns.

(F)RACHECK ATTR=	Hex Value	CA-Top Secret Default Access Level name	Hex Value
ALTER	80	ALL	FFFF
CONTROL	08	CONTROL	0400
UPDATE	04	UPDATE	8000
READ	02	READ	4000

Using the default values shown above, CONTROL access doesn't include UPDATE or READ. Therefore, to make the RACHECK value function hierarchically (that is, for CONTROL access to also include UPDATE and READ and for UPDATE to include READ), the following values must be explicitly defined in the resource class in the RDT.

Access Level	Hex Value
ALTER	FFFF
CONTROL	C400
UPDATE	C000
READ	4000

The following is a sample command used to implement the access levels in the preceding table.

```
TSS ADD(RDT) RESCLASS(SAMPLE) RESCODE(07)
      ACLST(ALTER(FFFF),CONTROL(C400),UPDATE(C000),READ(4000))
```

Access Level Placement: When assigning multiple levels, as the previous example does, the higher access level should be placed first, as shown below, because of the way CA-Top Secret lists and displays the PERMITTED access levels.

```
ACLST (ALL,CONTROL,UPDATE,READ)
```

Combining User-Defined and Standard Levels: You can also combine user-defined and standard access levels as shown in the example below.

```
ACLST (ALTER(FFFF),ALL,CONTROL,EDIT(8000),UPDATE,
VIEW(4000),READ,MOVE(0004))
```

Examples: The following example assigns unique access levels in which the list is non-exclusive. The DEFINE access level in this example allows all access levels that follow it—VIEW, EDIT, and MOVE. However, MOVE doesn't imply DEFINE, VIEW, or EDIT.

```
TSS ADD(RDT) RESCLASS(MENU) RESCODE(05)
ACLST(DEFINE(FFFF),EDIT(8000),VIEW(4000),MOVE(0400))
```

To further illustrate, suppose you assigned unique access levels to a main menu in the installation-written menu definition facility and you want to define new resource classes and access levels that are meaningful to your site. The following table shows the site-written access levels used to equal access levels passed by the ATTR= keyword of RACHECK.

Access levels	
Site Written	ATTR=
VIEW	READ
EDIT	UPDATE
DEFINE CREATE/DELETE	ALL or ALTER
MOVE	CONTROL

6.3.2.4 With the ACTION Keyword

Not all resource classes can be restricted by access level or path. For this reason, CA-Top Secret provides another layer of security using the ACTION keyword. The operand specified with the ACTION keyword tells CA-Top Secret how to respond to an access request. For example, the next command tells CA-Top Secret to notify the security console when USER03 accesses the PD000001 terminal.

```
TSS PER(USER03) TERM(PD000001) ACTION(NOTIFY)
```

ACTION operands include:

ADMIN	Allows a security administrator to administer resources that are not owned within his scope of authority. If an access level isn't specified, CA-Top Secret will PERMIT the default access level for that resource class.
AUDIT	Creates an audit trail when the resource is accessed, regardless of the mode or logging options of the user.
DENY	Denies the ACID access to the specified resource. The mode that applies to the user is still honored. ACTION(DENY) doesn't apply to resources where access levels are specified; instead, use ACCESS(NONE).
EXIT	Invokes the CA-Top Secret Installation Exit for all accesses to the specified resource granted by this permission.
FAIL	Treats any access attempt as if the user were in FAIL mode. In other words, CA-Top Secret will fail any unauthorized access attempt regardless of the security mode the facility or user is in.
NODSNCHK	Tells CA-Top Secret to only check volume authorizations for access requests to data sets on this particular volume.
NOTIFY	Issues a TSS7299W message to the security console on any access to the specified resource granted by this permission.
PASSWORD	Adds additional password protection before granting access to a data set. Specifically, returns control to VSE program checking for all DASD data sets after CA-Top Secret verification has authorized access to this data set. Note: Any data set checks that occur as a result of allocating an SMS-managed data set will not be prompted for a data set password.
VMPRIV	Grants the accessor the privileged form of CP commands and DIAGNOSE instructions.

6.3.3 Denying Access

You can selectively deny an ACID access to any resource.

- For a resource that doesn't support access levels (like a program, a terminal, or a CPU) or that you don't want users to access, use the ACTION(DENY) parameter on the PERMIT command function.
- For resources that support access levels, use the ACCESS(NONE) parameter on the PERMIT command function.

For example, assume that USER01 is connected to PROF01 which provides USER01 with several resource authorizations—including access to all the terminals in the Accounting Department. There are two terminals, however, that USER01 shouldn't be

allowed to access: (PD000001) and (PD000002). To remove authorization for USER01 from these terminals, enter:

```
TSS PER(USER01) TERM(PD000001,PD000002) ACTION(DENY)
```

Whether a request by USER01 to use either of these terminals is actually failed still depends upon the MODE setting. To ensure that this request is automatically failed regardless of the MODE setting, enter:

```
TSS PER(USER01) TERM(PD000001,PD000002) ACTION(DENY,FAIL)
```

These commands are particularly useful when PROFILE allows access which the security administrator wants to override.

6.4 The Security Validation Algorithm

Once all resources have been defined to CA-Top Secret and their access levels have been specified, any future request to access those resources is processed through the CA-Top Secret security validation algorithm. The security validation algorithm is the formula CA-Top Secret uses to determine whether an ACID has the appropriate authorization to access a particular resource. Many factors, such as the security mode, resource ownership, the order of permissions, and the use of the Installation Exit, are considered by the security validation algorithm.

Note: Your security mode settings control whether an authorization is checked, fails, or is accepted with a warning. Modes are discussed in Chapter 2, *Implementing CA-Top Secret*.

6.4.1 How Does the Security Validation Algorithm Work?

Whenever a user requests access to a particular resource, the Security Validation Algorithm is activated. The algorithm determines whether the requested access is granted by searching these records in the following order.

- the user record
- each attached profile in the order they were attached
- the ALL Record, if included in the search sequence

Whether or not access is granted depends on several factors:

- The value you have selected for the AUTH control option. This value controls how these records are searched, which records are included in the search, and when to stop the search.
- Whether you have specified the ATTR keyword for RESCLASS.
- Whether the request is for a volume, a data set, or another resource (called a *general* resource).

In addition, if multiple PERMITs for the resource are located, the PERMIT that most closely matches the request, called *best fit*, is selected by the algorithm.

The next sections explain how to set up your algorithm using the AUTH control option and the ATTR keyword, how requests for volumes, data sets, and general resources are processed, and how best fit is determined.

6.4.2 How to Set Up Your Security Validation Algorithm

The security validation algorithm consists of two components: The AUTH control option and the ATTR keyword of a resource class. The AUTH control option determines how the algorithm works and is always required. The ATTR keyword is used to fine-tune the search process by specifying how the search is performed for specific resources. The ATTR keyword is optional.

How to specify an AUTH control option value and how to use the ATTR keyword are covered in the next sections.

6.4.2.1 Specifying an AUTH Control Option Value

The value you choose for the AUTH control option specifies whether the user, profile, and ALL records are to be merged for the search or are to be searched as separate records. There are two sets of keywords that control how the search is conducted:

```

OVERRIDE|MERGE
ALLOVER|ALLMERGE

```

One value from the first set must always be selected. You can then select one value from the second set **if** you want to include the ALL Record in the search. As a result, you could have one of the following selections as your value for the AUTH control option:

```

OVERRIDE
MERGE
OVERRIDE,ALLOVER (the default)
MERGE,ALLOVER
MERGE,ALLMERGE

```

Note: OVERRIDE and ALLMERGE can't be specified together.

An explanation of each keyword appears below, followed by an explanation of how they work when specified together.

OVERRIDE Searches the user record and attached profiles sequentially, beginning with the user record and followed by each attached profile.

- Once an authorization is found for the resource in the user record, the search continues to the end of the user record. Profiles aren't checked since an authorization was located in the user record.
- If no authorization is located in the user record, profiles are searched in the order they were attached to the ACID. When an authorization is located, the rest of the profile containing it is searched and then the search stops. No other profiles are searched. Therefore, any authorizations that might exist in subsequent record(s) are ignored. Because of this, the order in which profiles have been added to the ACID is extremely important.

- If multiple PERMITs for the resource are located, access is granted or denied based on the PERMIT containing the resource prefix that most explicitly matches the resource being requested.

MERGE Searches the user record and profiles as if they were one contiguous record. An access decision isn't made until the entire merged record is searched. If no matching PERMIT is found, the ALL Record is searched for a match.

If multiple PERMITs for the resource are located, access is granted or denied based on the PERMIT containing the resource prefix that most explicitly matches the resource being requested. For example, if ACID USER01 tries to update data set ABC.DEF and the control option AUTH(MERGE) is in effect, both the user's primary and all attached profile security records will always be searched. If USER01 is also attached to profiles PROFA and PROFB, these two PERMITs are encountered during the validation search:

```
TSS PER(PROFA) DSN(ABC.) ACC(READ)
```

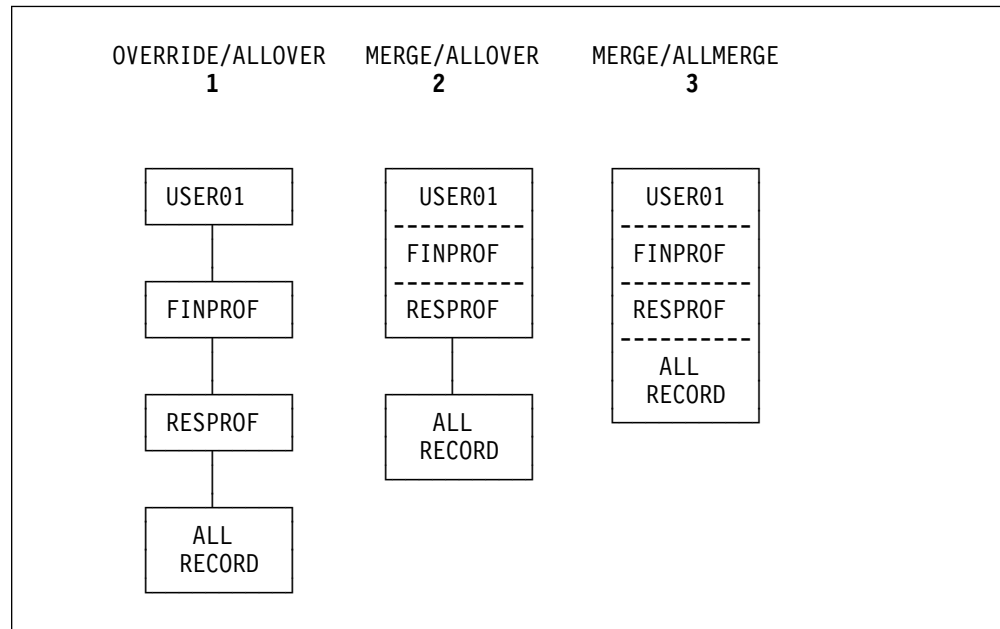
```
TSS PER(PROFB) DSN('ABC.DEF') ACC(UP)
```

Since the second PERMIT is a "better match" to the resource to which access is being requested, it will govern whether the request is granted. And, since it does provide authorization, the request is approved.

ALLMERGE Searches the user, profile, and ALL records as a single, merged record. All matches are evaluated. ALLMERGE can only be used with MERGE.

ALLOVER Adds the ALL Record to the designated search sequence, after all other records have been searched. ALLOVER can be used with either OVER-RIDE or MERGE.

To better understand how these options control the search, consider the following examples where USER01 has two attached profiles—FINPROF and RESPROF. The authorization he needs to access a particular resource may be in his user record, in one of the profiles, or in the ALL Record. Furthermore, there may be several PERMITs—each for a different type of access—for the same resource in one or more of those records.



1. **OVERRIDE** searches the user record (USER01) and each attached profile (FINPROF and RESPROF) sequentially. If **ALLOVER** is used with **OVERRIDE** the ALL record is also searched.
2. **MERGE** searches the user record (USER01) and all attached profiles as one record. If **ALLOVER** is used with **MERGE**, the ALL record is searched after the other records.
3. If **ALLMERGE** is used with **MERGE**, the ALL Record is added to the search and the user record, profiles, and ALL Record are searched as one record.

Processing Considerations: When selecting a value for the AUTH control option, you should keep these processing considerations in mind:

- **OVERRIDE** minimizes the number of I/Os by accepting the first match. This reduces the search time required to process permissions but may miss better matches in a later profile.
- Successively longer searches are needed for the **MERGE/ALLOVER** and **MERGE/ALLMERGE** options because the best match is found over all the records in the merge, rather than stopping the search on the first record where a match is found. However, with **MERGE/ALLMERGE**, the user is assured of the best match over the range of profiles attached to the user, regardless of the profile order within the processing record.

This tradeoff of processing time versus best fit should be considered when selecting your AUTH control option value or when overriding the AUTH value with a resource class attribute.

6.4.2.2 Using the ATTR Keyword

You can control how the search is performed for a particular resource by specifying one of the following values for the ATTR keyword of a resource class. When specified, the ATTR keyword value overrides the AUTH control option value for the resource class to which ATTR is attached. As a result, using the ATTR keyword is a way to set the AUTH control option value by resource class.

Value	Description
MERGE	Use the MERGE value for AUTH to check this resource class.
ALLMERGE	Use the ALLMERGE value for AUTH to check this resource class.
NOMERGE	Do not use the MERGE value for AUTH to check this resource class.
NOALLMERGE	Do not use the ALLMERGE value for AUTH to check this resource class.
MASK	Use the MASK value to support masking characters in a PERMIT.
NOMASK	Use the NOMASK value to turn off a previous MASK attribute.

For example, if the DATABASE resource class had the MERGE value specified for ATTR as shown below, whenever someone tries to access a database resource, the security validation algorithm will search for a PERMIT just as if MERGE had been specified for the AUTH control option.

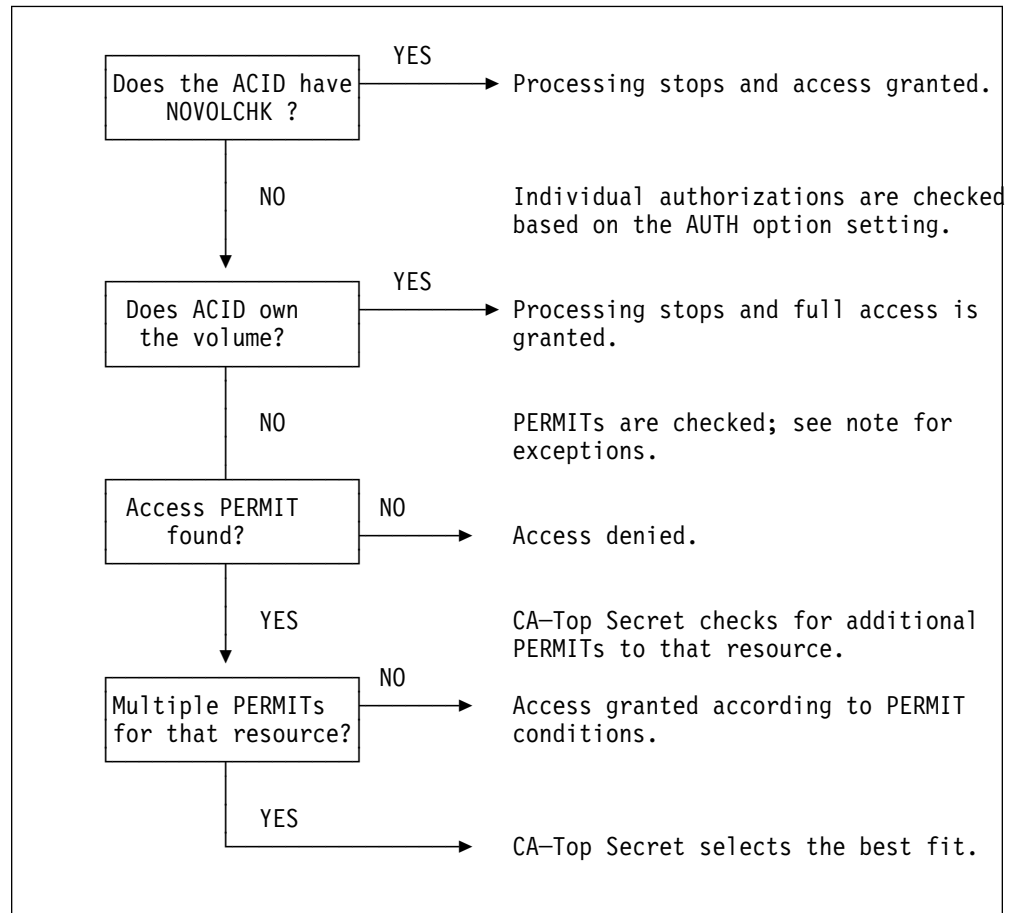
```
TSS REPL(RDT) RESCLASS(DATABASE) ATTR(MERGE)
```

6.4.3 Processing Volume Requests

When validating access to a volume, either for a volume check or as part of a data set check, the Volume Call is invoked first. This call grants or denies access to the volume or data set, or allows normal processing to continue. (Volume access overrides data set access as shown in the following chart.) Before processing continues, CA-Top Secret makes the following inquiries about the records being searched:

- Does the ACID have the NOVOLCHK attribute?
- Does the ACID own the volume?
- Was an access PERMIT found?
- Were multiple PERMITs found for the requested resource?

How these inquiries are handled is shown in the following flowchart.



Note: A PERMIT is not checked if it:

- Has an access level of BLP when the access request is for a DASD volume of a data set.
- Has an access level of CREATE, SCRATCH, or CONTROL if the access request is for a tape volume.

6.4.4 Processing Data Set Requests

Data set checking is performed when a DSN has been opened for TAPEVOL calls when the TAPE(DSN) control option is in effect. When an ACID requests access to a particular DASD data set, the pertinent volume and data set access authorizations must be evaluated depending on what bypass options are associated with that ACID or PERMIT. The bypass options include whether or not the ACID has NODSN access for that specific request, and whether or not that ACID has been given the NODSNCHK or NOVOLCHK attributes.

Note: Generally, for tapes, either volume level or data set level security is in effect depending upon whether or not you are using a tape management package.

When both volume and data set level access checking are being done, CA-Top Secret always performs volume level first. In some cases a request to access a data set is granted or failed strictly on the basis of the ACID's volume access authorizations—without checking whether he has specific authorization to access that particular data set. Table 6-4 shows how volume access authorizations affect an ACID's request to access a data set on that volume. The authorized volume access is listed in the first column; the attempted data set access is listed in columns two through four.

Authorized Volume Access	Data Set Read	Data Set Update	Data Set Create	Data Set Scratch
READ	OKAY	DSN	FAIL	DSN
UPDATE	OKAY	OKAY	FAIL	DSN
CREATE	DSN	DSN	DSN	DSN
NOCREATE	DSN	DSN	FAIL	DSN
ALL	OKAY	OKAY	OKAY	OKAY
NONE	FAIL	FAIL	FAIL	FAIL
SCRATCH	DSN	DSN	FAIL	OKAY
UNDEFINED	DSN	DSN	FAIL	DSN

* UNDEFINED means that the ACID making the request has no relevant access authorizations for that volume.

Key:

OKAY Access will be granted.

FAIL Access will be denied.

DSN Access will depend on what data set name security is permitted.

Note: If the user has both CREATE and CONTROL access, data set creation will be performed regardless of the data set name.

To further illustrate the volume/data set relationship shown in the previous table, consider what would happen if an ACID with UPDATE access to a volume requested access to a data set on that volume.

- A request for READ and UPDATE access to those data sets would be granted immediately (OKAY).
- A request for CREATE access would be denied immediately (FAIL).
- A request for SCRATCH access would cause CA-Top Secret to check the ACID's data set authorizations (DSN).

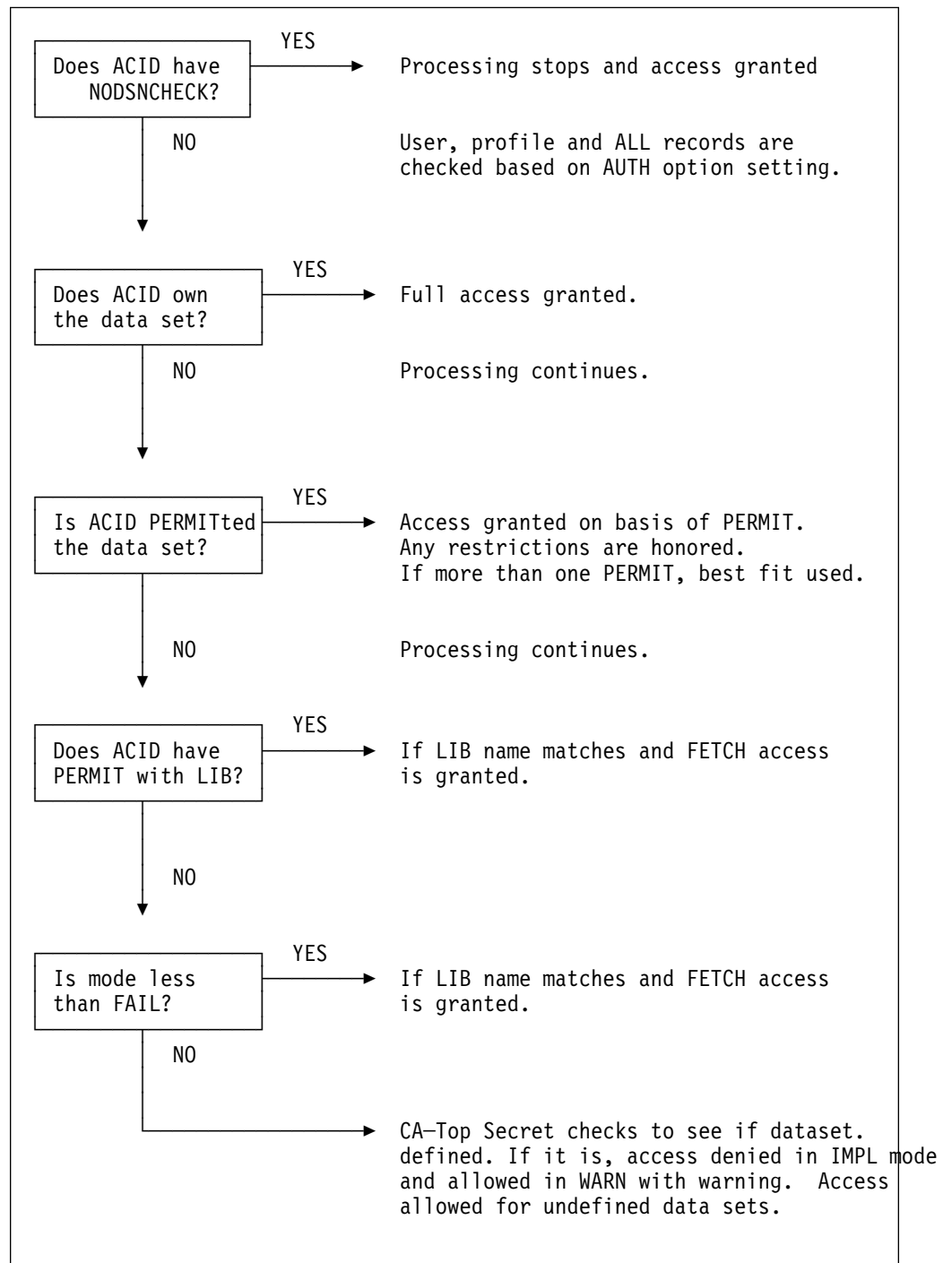
Table 6-3 will only be used for requests where both volume and data set authorizations are checked. For cataloging a tape data set, a CREATE access check will be made only at the data set level. This is independent of the setting of the TAPE control option. No volume check will be made, so even having ALL access at the volume level will not allow a tape data set to be catalogued. Also, when allocating an SMS-managed data set, no volume is identified on the data set call and no volume checking takes place.

If a VOLUME check that was qualified by PRIVPGM, FACILITY, DATE, or TIME denies access on a data set request, this rule of data set authorization relationships will be honored, rather than continuing to the DATASET name checking.

The Data Set Call: Data set checks occur after volume access has been satisfied. To perform data set checking, the Data Set Call is invoked. The data set call is then granted or denied. Before processing continues, CA-Top Secret makes the following inquiries about the records being searched:

- Does the ACID have the NODSNCHK attribute?
- Does the ACID own the data set?
- Is the data set owned by another ACID?
- Is the ACID permitted to the data set?
- Are there restrictions on the permissions?
- Does the PERMIT to the data set have PRIVPGM with LIB specified?
- What is the mode?

How these inquiries are handled is shown in the next flowchart.



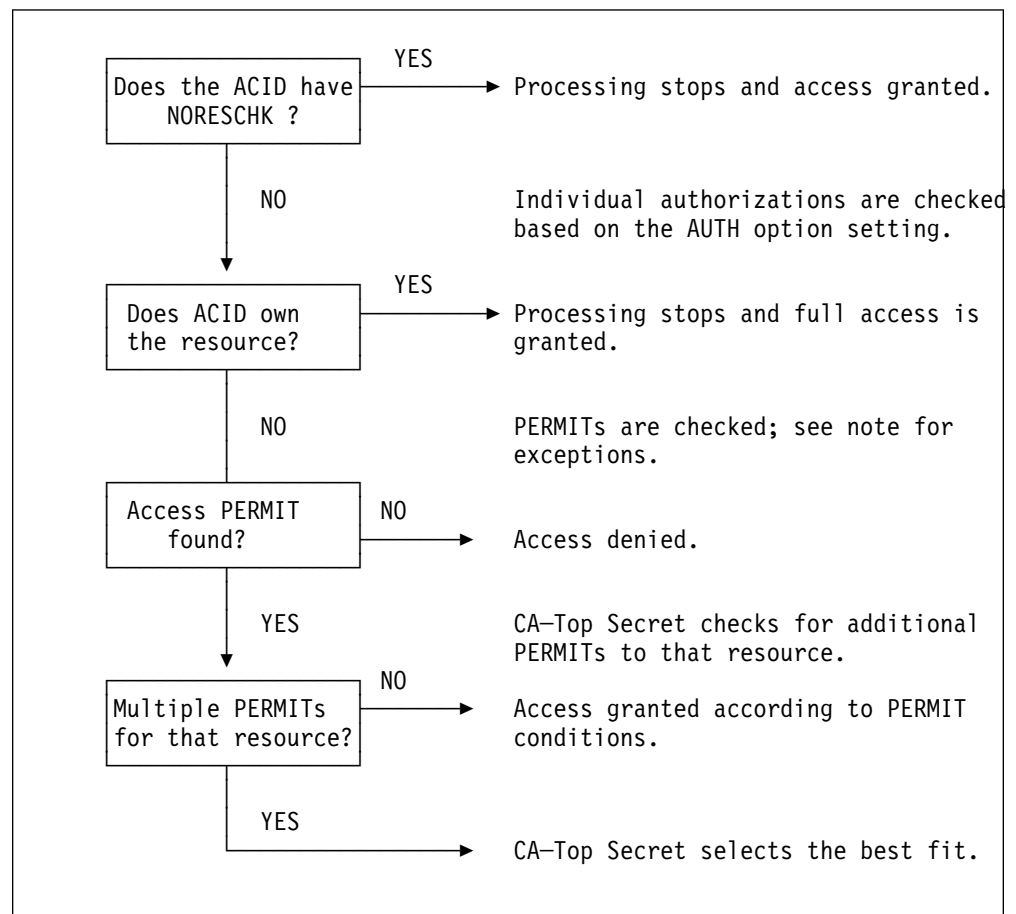
6.4.5 Processing General Resource Requests

When a user requests access to a particular resource, CA-Top Secret access authorizations are determined using the security validation algorithm. When multiple PERMITs are found, access is determined by "best fit", rather than by the most restrictive match.

As with data set and volume checking, CA-Top Secret makes the following inquiries about the records being searched:

- Does the ACID have the NORESCHK attribute?
- Does the ACID own the resource?
- Was an access PERMIT found?
- Were multiple PERMITs found for the resource?

How these inquiries are handled is shown in the next flowchart.



6.4.6 How Best Fit is Determined

When multiple PERMITs are located during the search, the PERMIT that most closely matches the request is selected according to a set of criteria. The first two criteria used are always the same for all resource types (volumes, data sets, and general resources).

- The first criteria is the longest resource name specified. When multiple PERMITs are located, the one containing the longest resource name is considered the best match.
- If the PERMIT is valid but the access level for the resource is NONE, then access is denied.

Determining Longest Resource Name: For determining the longest resource name, each character in the resource name counts as **one** whether it is a normal character or a masking character. However, if the floating mask "-" is used in a PERMIT, only the characters prior to the mask are counted. The following table contains examples of how characters are counted.

For Resource:	Number of characters counted:
SMITH.TEST.JCL	14, all characters are counted
JONES.*.JCL	11, all characters are counted
BROWN.-.JCL	6, only 'BROWN.' is counted

If best fit can't be determined by resource name length or an access level of NONE, then the criteria used to determine the best fit differs for volumes, data sets, and general resources.

Criteria for Volumes: The criteria used to determine best fit for volumes are:

- A valid PERMIT with full access to the volume is considered a match.
- A volume PERMIT with an access level of CREATE is considered a match when the request was for data set CREATE (data set checking will follow).
- The first violation due to access level mismatch denies access.
- The first violation for any other reason (for example, DAY, TIME, FACILITY or PRIVPGM mismatches) denies access.

Criteria for Data Sets: The criteria used to determine best fit for data sets are:

- PERMITs using quotes (i.e. 'A.B.C.') are considered matches.
- PERMITs without quotes, but not starting with a masking character would be considered matches.
- PERMITs starting with a mask would be considered matches.
- A valid PERMIT with an access level of ALL would be considered a match.
- The first violation against a FETCH-protected data set denies access.

- The first access level mismatch denies access.
- The first violation for any other reason (for example, DAY, TIME, FACILITY, or PRIVPGM mismatches) denies access.

For General Resources: The criteria used to determine best fit for other resources are:

- The first violation due to access level mismatch denies access.
- The first violation for any other reason (for example, DAY, TIME, FACILITY, or PRIVPGM mismatches) denies access.

Chapter 7. Protecting Resources

CA-Top Secret supports the protection of many types of computer resources. The types of resources (such as data sets, volumes, and terminals) that CA-Top Secret protects are listed in the Resource Descriptor Table (RDT). Many resource types are already automatically defined to the RDT at installation; however, additional resource types (including site-defined resources) can be added by using the TSS ADD command function.

CA-Top Secret can also protect access to products and services in use at your site. It can protect individual commands, tasks, or transactions connected with these services. It can protect several resources used by jobs, tasks, and services throughout the VSE system. The following lists show examples of these resource types:

Facilities	Resources	Commands
VM	Data Sets	TSO commands
BATCH	LIBR members	
TSO	Programs	CA Product commands
CICS	Terminals	DL/1 commands
CA-IDMS	Readers	
TSO (Started Tasks)	Nodes	
DL/1	User-defined resources	
User-defined	Operator commands	
User-defined	DASD volumes	
	Tape volumes	
	CICS resources	
	DL/1 resources	
	CA-IDMS resources	
	POWER resources	

In general, resource protection is a two step process.

Step 1 Assign ownership.

Most resources can be specified either by their full names or by generic prefixes when assigning ownership.

Step 2 Authorize permissions.

Once a resource is owned, you must authorize its use by other ACIDs.

This section contains detailed instructions for protecting these resources:

- Volumes
- Data sets
- Library members
- Tape volumes
- Terminals and Consoles
- Programs
- Applications
- Installation-defined resources
- Commands
- POWER resources

Bypassing Resource Checking: In rare cases, you may want to extend the bypassing privilege so that any time a particular ACID makes a certain type of access request, those requests will bypass security checks. In CA-Top Secret this is done by adding one or more bypass attributes to the ACID.

Note: This type of "blank check" for security access isn't recommended for ordinary use, but may be considered when setting up disaster recovery procedures.

The following list of bypass options (also called no-check attributes) should be used with great care, if at all. They are:

NORESCHK Allows an ACID to bypass all resource checking except for data sets and volumes.

NOVOLCHK Allows an ACID to bypass all volume checking.

Note: If data set access is being requested, CA-Top Secret responds according to the data set authorizations. To allow an ACID unrestricted access to an **entire** volume, you must also add the NODSNCHK attribute.

NODSNCHK Allows an ACID to bypass data set checking. Don't confuse this attribute with ACTION(NODSN), which only affects access requests for a particular volume.

NOLCFCHK Allows an ACID to bypass LCF (Limited Command Facility) checking. For more details on LCF security, refer to 7.9.3, "Implementing LCF Security" on page 7-39.

NOSUBCHK Allows an ACID to bypass job submission security checking. With this attribute, an associated ACID can submit all jobs regardless of the derived ACID on the job card being submitted.

NOVMDCHK Allows an ACID to bypass VM minidisk link checking.

These attributes are added to an ACID through the TSS ADD command function, and revoked through the TSS REMOVE command function. For example, the following statement allows the SUPRACID to bypass **all** resource checking.

```
TSS ADD(SUPRACID) NORESCHK NOVOLCHK NODSNCHK NOVMDCHK
```

To remove the privileges, use the statement shown below.

```
TSS REM(SUPRACID) NORESCHK NOVOLCHK NODSNCHK NOVMDCHK
```

Using NORESCHK: To bypass security checking for all resources except data sets and volumes, attach the NORESCHK attribute to a particular ACID or profile. Specifying NORESCHK also bypasses any auditing that would otherwise be performed. NORESCHK is a powerful attribute; therefore, its use should be carefully restricted. An example showing how to specify this attribute appears below.

```
TSS ADD(SUPRACID) NORESCHK
```

To remove a user's bypass privilege, use the TSS REMOVE command function as shown below.

```
TSS REM(SUPRACID) NORESCHK
```

7.1 Protecting Volumes

Like other ownable resources, volumes (both DASD and tape) must first be owned before being authorized.

Note: Tape volumes and their management is discussed in 7.3, “Protecting Tape Volumes” on page 7-18.

How to Add Ownership: To establish ownership, use a TSS CREATE/ADD VOLUME entry like the one shown below; then use a TSS PERMIT VOLUME entry to specify authorizations. The following example protects the volume whose VOL=SER is T64803 by assigning ownership of the volume to DEPT01.

```
TSS ADD(DEPT01) VOL(T64803)
```

Note: Another option is to bypass volume level security checking entirely and depend upon data set level security checking instead. Refer to 7.1.4, “How to Bypass Volume Level Security” on page 7-8 for details.

Once a volume has been protected, an access request to the volume or to any data set on the volume needs access to the volume. See Table 6-4 on page 6-42 for an explanation of how volume and data set access levels are related.

How to Remove Ownership: Removing ownership of a volume is a two step process.

Step 1 Revoke all permissions for the resource. A sample command appears below.

```
TSS REV(USER01) VOL(T64803)
```

You can't specify an access level or the command will fail.

Tip: Use TSS WHOHAS to find out who has PERMITs for the resource.

Step 2 Remove the ownership of the volume. A sample command appears below.

```
TSS REMOVE(DEPT01P) VOL(T64803)
```

Note: CA-Top Secret won't remove ownership unless all permissions are revoked.

7.1.1 How to Extend Default Protection

Default protection extends security protection to a volume even if that volume hasn't yet been defined to CA-Top Secret. As a result, a security violation will occur if a request is made to access any unowned volume. If this type of request occurs in FAIL mode, the request is denied.

To extend default protection for volumes, attach the DEFPROT attribute to the resource class named VOLUME using the TSS REP(RDT) command function as shown below.

```
TSS REP(RDT) RESCLASS(VOLUME) ATTR(DEFPROT: ehp2.)
```

7.1.2 How to Use Generic Prefixing

The VOLUME resource is installed with the NONGENERIC attribute. Nevertheless, tape or DASD volume ownership can be designated with generic prefixes. Once a prefix is owned, any volume beginning with that prefix is protected and must be permitted to other ACIDs.

The maximum length allowed for a volume resource identifier is six characters for a specific volume and five characters for a generic prefix. The minimum length allowed for the generic prefix is two characters. When using generic prefixing, you must append a G to the prefix as shown in the next example to indicate that a generic prefix is being supplied.

```
TSS ADD(DEPT01) VOL(T64(G))
```

This example protects all volumes with T64 as the first three characters of their VOL=SER.

Generic prefixing is discussed in detail in Chapter 6, *Resource Security Validation*.

7.1.3 How to Authorize Access

The PERMIT command function is used to allow designated users to use the specified volumes in either an unlimited or restricted manner. Restrictions are specified by incorporating the appropriate PERMIT parameters. For example, the following PERMIT allows USER01 to access volume T64803 from Monday through Friday to create data sets.

```
TSS PER(USER01) VOL(T64803)
      DAYS(WEEKDAYS) ACC(CREATE)
```

Commonly used restrictions (like time, duration, and facility) are discussed in Chapter 5, *Password and System Entry Security*. For a complete list of the parameters that can be used with PERMIT see the *Command Functions Guide*.

Volume level access allows a user to access any data set on the volume, provided he has the authorized level. Access levels for volumes are discussed in the next section.

7.1.3.1 How to Assign Access Levels

Access levels are assigned using the ACCESS parameter. This parameter is used to restrict how volumes can be accessed. The following list contains the access levels that can be specified for volumes; READ is the default.

Level	Meaning
ALL	All data residing on a volume can be accessed in any way.
BLP	Tape volume can be used with Bypass Label Processing.
UPDATE	All data sets residing on a volume can be simultaneously opened for both read and write access; implies READ volume authorization. This access level is needed to perform a restore of the volume.
READ	All data sets residing on a volume can be read (opened for input); the default. This access level is needed to perform a backup of the volume.
WRITE	Volume can only be opened for output.
CREATE	Data sets can be created on this volume if the data set access level permits it.
SCRATCH	Any data sets residing on a volume can be scratched.
CONTROL	If included with CREATE access, the user can create any data set residing on a volume—regardless of applicable data set access authorization.
NOCREATE	All data sets residing on a volume can be accessed according to their data set access level. However, no new data sets can be created on this volume, regardless of the user's data set authority.
NONE	Data residing on a volume can't be used in any way.

It is recommended that volumes should rarely be permitted for more than CREATE access to individual users, since the volume authorization generally overrides the data set authorization. Typically, the more powerful access levels should be reserved for DASD management functions.

The example shown below authorizes USER01 to update any data set on any volume whose VOLSER begins with T64.

```
TSS PER(USER01) VOL(T64(G)) ACC(UPDATE)
```

Note: If a user possesses no specific volume authorization, he can't create data sets on that volume regardless of his data set authorization.

How CA-Top Secret acts in situations in which both volume and data set access authorizations apply is discussed in 7.4, “Data Set Vs. Volume Level Security” on page 7-20.

7.1.3.2 How to Assign a Program Path

Program pathing can be used to restrict the access of certain volumes to designated programs. Program pathing can be an especially useful tool for enforcing security protection when a volume management package is being used. Extensive access to volumes can be allowed, but only through the appropriate volume management programs. For example, the following PERMIT gives the ACID VOLMGR access to all volumes at an access level of ALL, but only through the program VMGRP01.

```
TSS PER(VOLMGR) VOL(*ALL*(G)) ACC(ALL)
PRIVPGM(VMGRP01)
```

Program pathing is discussed in detail in Chapter 6, *Resource Security Validation*.

7.1.3.3 How to Access All Volumes

To allow users access to all volumes—DASD as well as tape—use the resource identifier *ALL*(G). Before you can use this identifier, you must assign ownership of it to the MSCA, as shown in the following example.

```
TSS ADD(MSCA) VOL(*ALL*(G))
```

After ownership has been assigned, you can use the identifier in a PERMIT statement as shown below.

```
TSS PER(USER01) VOL(*ALL*(G))
```

This example allows USER01 to read any volume.

7.1.4 How to Bypass Volume Level Security

There are two methods you can use to bypass all volume level security:

1. Using the ***ALL*(G)** indicator (the recommended method).
2. Attaching the **NOVOLCHK** attribute to a particular **ACID** or profile.

When volume level security is bypassed, the CA-Top Secret response to a request to access a data set is based solely on the applicable data set authorization (see 7.2.4, “How to Authorize Access” on page 7-13).

Using *ALL*(G): To use the ***ALL*(G)** indicator to bypass volume level security, you must first assign ownership of it to the **MSCA**, as shown below.

```
TSS ADD(MSCA) VOL(*ALL*(G))
```

Once ownership has been assigned, authorize permission using the **PERMIT** function shown below.

```
TSS PER(USER01) VOL(*ALL*(G)) ACCESS(CREATE)
```

Using NOVOLCHK: **NOVOLCHK** is a powerful attribute and cannot be traced with **TSS WHOHAS**; therefore, its use should be restricted to special products, such as volume management packages, that must access many volumes. Another restriction includes online regions which dynamically allocate data sets such as **CA-Roscoe** or **CA-IDMS**. An example showing how to specify this attribute appears below.

```
TSS ADD(SUPRACID) NOVOLCHK
```

Note: To allow unrestricted access to an entire volume, whether there are any defined data sets on it or not, specify the **NODSNCHK** attribute with the **NOVOLCHK** attribute.

7.1.5 How to Request Only Volume Level Security

You can bypass data set level security checking by attaching the keyword **ACTION(NODSN)** to a **PERMIT** for a volume. When this parameter is specified, CA-Top Secret considers only the pertinent volume authorizations when determining whether to grant an access request. For example, the **PERMIT** shown below allows **USER01** to scratch (but only scratch) any data set on volume **T50000**.

```
TSS PER(USER01) VOL(T50000)
      ACTION(NODSN) ACC(SCRATCH)
```

Unlike the BYPASS attribute NOLVOLCHK, the ACTION parameter operates at the level of a specific access event—that is, an access request made by a designated ACID for a designated resource.

7.1.6 Using Volume Management Packages

The following CA, IBM, and vendor volume management products interface automatically with CA-Top Secret:

```
CA-ASM/ARCHIVE
CA-Dynam
```

7.1.6.1 Required Backup and Restore Access Authorizations

The following access level authorizations are generally required for the common volume utility procedures backup, restore, and compact.

Procedure	Access Level
backup	READ
restore	UPDATE
compact	ALL

7.2 Protecting Data Sets

Like other ownable resources, data sets (both DASD and tape) must first be owned before being authorized. To establish ownership, use a TSS CREATE/ADD DSN entry like the one shown below; then use a TSS PERMIT DSN entry to specify authorizations as shown in 7.2.4, “How to Authorize Access” on page 7-13.

```
TSS ADD(DEPT01) DSN(PAYROLL.)
```

Note: Another option is to bypass data set level security checking entirely and depend upon volume level security checking instead. Refer to Bypassing Data Set Checking From the Volume on page 7-17 for details.

How to Remove Ownership: Removing ownership of a data set is a two step process.

Step 1 Revoke all permissions for the resource. A sample command appears below.

```
TSS REV(USER01) DSN(PAYROLL.)
```

You can't specify an access level or the command will fail.

Tip: Use TSS WHOHAS to find out who has PERMITs for the resource.

Step 2 Remove the ownership of the data set. A sample command appears below.

```
TSS REMOVE(DEPT01) DSN(PAYROLL.)
```

Note: CA-Top Secret won't remove ownership unless all permissions are revoked.

7.2.1 How to Extend Default Protection

CA-Top Secret provides default protection of data sets in FAIL mode. Default protection extends security protection to a data set even if it hasn't yet been defined to CA-Top Secret. As a result, a security violation will occur if a request is made to access any unowned data set, and the request is denied.

To extend default protection to data sets in IMPLEMENT and WARN modes, attach the DEFPROT attribute to the resource class named DATASET using the TSS REP(RDT) command function as shown below.

```
TSS REP(RDT) RESCLASS(DATASET) ATTR(DEFPROT)
```


Recommendations: In IMPLEMENT mode, in general, it is best to give data sets default protection. This preserves the concept of using IMPLEMENT mode to bring an installation under the CA-Top Secret security umbrella in controlled phases. In addition, in a non-Alwayscall environment it is generally best to set ADSP(ALL) (see the previous section) to prevent undefined users from being able to create data sets anywhere and then access them without security checking.

7.2.2 How to Use Generic Prefixing

Data set ownership is typically designated by installation default using generic prefixes. Once a prefix is owned, any data set beginning with that prefix is protected and must be permitted to other ACIDs. The following example gives ownership of system data sets that begin with the prefixes SYS or IPO to the ACID SYSGROUP.

```
TSS ADD(SYSGROUP) DSN(SYS,IPO)
```

Generic prefixing is discussed in detail in Chapter 6, *Resource Security Validation*.

7.2.3 How to Use Masking

Because data sets aren't only primary resources but are also the most numerous of the resource types, you can reduce the number of entries that must be made to secure your site's data sets by using masking (also called patterning). Masking can be used to group data sets whose names share similar characteristics. These shared patterns can then be used as the operands of the DSN parameter in TSS entries.

A masked data set name is treated by CA-Top Secret like a generic prefix. Any data set that begins with a mask is considered a match by the security validation algorithm, and the associated access authorizations are honored. However, you can add a resource name containing a mask only if the name begins with a masking character. For example, you can issue:

```
TSS ADD(dept01) DSN(+bc.+yz)
```

but not

```
TSS ADD(dept01) DSN(abc.+yz)
```

To use masking in the last example, you must first add the high level qualifier of the resource name and then PERMIT the resource name with a masking character as shown below.

```
TSS ADD(dept01) DSN(abc)
TSS PER(user01) DSN(abc.+yz)
```

Note: Masking characters can't be used within the owned portion of the DSN. For example, if TSS ADD(DEPT01) DSN(ABC.XYZ) is issued to establish ownership for data set abc.xyz, then TSS PER(ACID) DSN(ABC.X*) isn't valid.

Masking is discussed in detail in Chapter 6, *Resource Security Validation*.

7.2.4 How to Authorize Access

Use the TSS PERMIT command function to allow designated users to access the indicated data sets in either an unlimited or a restricted manner. Restrictions are indicated by incorporating the appropriate PERMIT parameter. For example, the following PERMIT allows USER01 to read data sets beginning with the qualifier PAYROLL. through the Batch facility.

```
TSS PER(USER01) DSN(PAYROLL.)
      ACCESS(READ) FAC(BATCH)
```

To specify a particular data set, place single quotes around the full data set name as shown next. This example authorizes USER01 to have complete access to the data set TNT.LIB.CNTL.BOOM.

```
TSS PER(USER01) DSN('TNT.LIB.CNTL.BOOM') ACC(ALL)
```

Commonly used restrictions (such as time, duration, and facility) are discussed in Chapter 5, *Password and System Entry Security*. For a complete list of the parameters you can use with PERMIT see the *Command Functions Guide*.

7.2.4.1 How to Assign Access Levels

Access levels are assigned using the ACCESS parameter. This parameter is used to restrict how data sets can be accessed. The following list shows the access levels that can be specified for data sets (VSAM and non-VSAM); READ is the default. (Access levels aren't presented hierarchically.)

Note: If a data set authorization is omitted on the TSS PERMIT command function, a default access level of READ is assigned.

Level	Meaning
ALL	Data set can be accessed in any way.
UPDATE	Data set can be updated; READ and WRITE access is implied. This access level is needed to perform a restore of the data set.
READ	Data sets can be read (opened for input); the default. READ implies FETCH. This access level is needed to perform a backup of the data set.
WRITE	Data can only be written into the data set (opened for output).
CREATE	Data set can be created.
FETCH	Programs from the data set (library) can only be executed, not read.
SCRATCH	Data set can be scratched.
CONTROL	VSAM data set can be used for control interval update processing (for example, for an IDCAMS VERIFY function). VSAM data set that had a control password can be used. For details, see Access Levels for VSAM Data Sets on page 7-14. For non-VSAM data sets, this access level is equivalent to WRITE access.
NONE	Data set can't be used in any way.

The example shown below authorizes USER01 to update any data set that matches the masked prefix SAL*M.

```
TSS PER(USER01) DSN(SAL*M) ACC(UPDATE)
```

If the CA-Top Secret security validation algorithm detects two data set name permissions of equal length, it selects the first permission with a PERMIT access level of NONE (if such a permission exists). Otherwise, the first PERMIT with any access level is selected. Refer to 6.4, “The Security Validation Algorithm” on page 6-36, for more information.

Access Levels for VSAM Data Sets: To access VSAM data sets the access level required corresponds to the previous password level for the catalog:

- READ access when READ password was used
- UPDATE access when UPDATE password was used
- ALL access when MASTER password was used

Access Levels for Creating Data Sets: To create a data set, the ACID must meet one of these conditions:

1. Own the volume.
2. Possess CREATE (or ALL) access at both the volume and data set level.

Note: CA-Top Secret can't prevent undefined users from creating data sets in a VSE environment that is non-Alwayscall except in FAIL mode. This is because VSE doesn't recognize CA-Top Secret modes and VSE controls new data set creation for defined users only.

3. Possess CREATE and CONTROL at the volume level.

7.2.4.2 How to Assign a Program Path

Program pathing can be used to restrict access to sensitive data to designated programs. Program pathing is an excellent way to:

- Restrict production control personnel from accessing inappropriate data without keeping them from performing their regular functions.
- Block any unwanted exposure introduced by CA-Panvalet—namely, always opening programs for UPDATE access even if only READ access is needed. Using program pathing, you can force access to the data set through a program that only has the ability to read it.

Simple Program Pathing: Program pathing is implemented with the PRIVPGM parameter. The example shown below authorizes USER01 to read data sets whose highest level qualifiers are SALPAY.MASTER, but only through program APUPDATE when running in the Batch facility.

```
TSS PER(USER01) DSN(SALPAY.MASTER)
PRIVPGM(APUPDATE) FAC(BATCH)
```

In addition, the program if APUPDATE is protected, the following PERMIT is also required.

```
TSS PER(USER01) PROG(APUPDATE)
```

7.2.4.3 How to Access All Data Sets

To allow users access to all data sets you can use the variable character substitution mask DSN(*****). However, before using this mask you must assign ownership of it to the MSCA, as shown in the next example.

```
TSS ADD(MSCA) DSN(*****)
```

After ownership has been assigned, you can use the mask in a PERMIT statement as shown below. ALL access is assigned to bypass data set level security, giving USER01 complete access to any data set.

```
TSS PER(USER01) DSN(*****) ACC(ALL)
```

There are several advantages to using this approach rather than the NODSNCHK attribute to bypass data set level security checking. They are:

- All the restrictions available to the PERMIT command function (such as time of day and privileged program pathing) can also be specified.
- You have the option of granting a less powerful access level, such as READ or SCRATCH.
- A TSS WHOHAS DSN(*****) can be used to find which users have access to all data sets.
- Data set auditing will take place if ACTION(AUDIT) is specified for the ACID or the data set being accessed is in the Audit Record.

7.2.5 How to Bypass Data Set Level Security

You can bypass all data set level security by attaching the NODSNCHK attribute (also called a BYPASS attribute) to a particular ACID or profile. However, volume level security checking is still in effect and auditing still occurs. An example appears below.

```
TSS ADD(SUPRACID) NODSNCHK
```

NODSNCHK is a powerful attribute; therefore, its use should be restricted to special products, such as DASD space managers, that must access many data sets.

Bypassing Data Set Checking From the Volume: You can bypass data set level security checking by attaching the keyword ACTION(NODSN) to a PERMIT for a volume. For more details on this procedure, see 7.1.5, “How to Request Only Volume Level Security” on page 7-8.

7.3 Protecting Tape Volumes

The type of tape volume protection that CA-Top Secret provides is specified through the TAPE control option and is dependent upon whatever tape management system is in use at your installation.

Note: A catalog or external tape management package listing can be used to identify most of your installation's volumes and their users.

The TAPE control option has three different settings:

TAPE(OFF) VSE won't invoke CA-Top Secret to validate a tape access request. This setting is generally appropriate when an external tape management package, such as Dynam/T, is in place at your installation.

TAPE(DEF) CA-Top Secret validates access requests for defined tape volumes only.

TAPE(DSN) CA-Top Secret validates tape data set access requests based on the full data set name specified in the //TLBL parameter of the JCL. (VSE limits the length of a data set name on a tape label to its last 17 characters.) With this option, tape security functions at the data set rather than the volume level.

7.3.1 Using a Tape Management Package

Any tape management package that issues CAISSF or SAF calls automatically supports CA-Top Secret at the tape volume and/or tape data set level. When used with tape management packages, CA-Top Secret can also be used to manage scratch tapes, synchronize tape inventories, and remove and assign tape ownership.

CA-Top Secret interfaces with most vendor tape management packages. If your installation is using CA-DYNAM/T, CA-Top Secret can provide full tape data set security since CA-Top Secret interfaces with this package at the data set level. If you aren't using a tape management package, then CA-Top Secret can protect your tape data set with certain restrictions. For more information about these restrictions, refer to your *Implementation Guide*.

Note: Because of the limitations on tape data set security in the absence of a tape management package, you can use CA-Top Secret to protect tapes at the volume level. It is recommended that you use a tape security package, such as CA-DYNAM/T.

7.3.2 Securing Bypass Label Processing in OS/390

A central security administrator can authorize designated ACIDs to run jobs that use bypass label processing (BLP). (Special initiators don't have to be set up to run BLP jobs.) Correspondingly, an ACID is not allowed to use BLP processing unless explicitly authorized.

Authorization to use BLP can be given for a specific volume, a generically defined group of volumes, or all tape volumes. This authority can be for update-level access or restricted to read-only access. To assign BLP authority use the following model.

```
TSS PER(acid) VOL(volser) ACCESS(BLP,READ or UPDATE)
```

For example, to give USER01 BLP processing authority to update any tape volume enter:

```
TSS PER(USER01) VOL(*ALL*(G)) ACC(BLP,UPDATE)
```

7.4 Data Set Vs. Volume Level Security

To determine whether access to a particular DASD data set should be allowed, CA-Top Secret must often evaluate both the pertinent volume and data set access authorizations.

Note: For tape volumes, generally either volume or data set level security will be in effect depending on the absence or presence of a tape management package.

CA-Top Secret **always** performs volume level checking first. In some instances a request to access a data set is evaluated strictly on the basis of the volume access authorizations located by the volume level check. For example, if the user has been authorized for any volume level access other than CREATE and the request doesn't "exceed" this access level, then CA-Top Secret allows access to the volume—and therefore, the data set—without checking for pertinent DSN authorizations. Also, if the ACID owns the volume, access is allowed without any data set validation. Correspondingly, if the user's volume access authorization is NONE, the request to access a protected data set on that volume is immediately denied.

If the ACID has no relevant VOLUME authorization, CA-Top Secret immediately switches to data set (DSN) validation checking (unless this is a data set creation request, in which case the request is automatically failed). For details on how data set requests are processed, see the section *Processing Data Set Requests* in Chapter 6, *Resource Security Validation*.

7.5 Protecting Terminals

This section explains how to secure terminals by using TSS commands to define the different types of terminals by ensuring that you have the appropriate attributes and defaults in the Facility Matrix.

The following terminal and reader device types can be defined to CA-Top Secret so that access to them has to be explicitly authorized:

- online terminals (VTAM and BTAM)
- remote (RJE) and local POWER readers
- internal readers
- NJE nodes

You can enforce two types of terminal protection:

1. Protect against unauthorized use of a protected terminal.

In general, this type of terminal protection is most useful with readers and RJE stations that don't require signons. For such devices, ordinary password security may not be feasible. However, by defining the reader to CA-Top Secret, access to this device can be controlled. This includes not only user access, but restricting which jobs are eligible for execution from this device through TSS ADD ACID entries.

2. Force user initiation only from an authorized source terminal.

This type of protection can be used to channel sensitive activities through authorized pathways.

Like other ownable resources, terminals must first be owned before being authorized. To establish ownership, use a TSS CREATE/ADD TERMINAL entry like the one shown below; then use a TSS PERMIT TERMINAL entry to specify authorizations. The following example protects all terminals whose names begin with K18L by assigning ownership of them to DEPT01.

```
TSS ADD(DEPT01) TERM(K18L)
```

How to Remove Ownership: Removing ownership of a terminal is a two step process.

Step 1 Revoke all permissions for the resource. A sample command appears below.

```
TSS REV(USER01) TERM(K18L1125)
```

Tip: Use TSS WHOHAS to find out who has PERMITs for the resource.

Step 2 Remove the ownership of the terminal. A sample command appears below.

```
TSS REMOVE(DEPT01P) TERM(K18L1125)
```

Note: CA-Top Secret won't remove ownership unless all permissions are revoked.

7.5.1.1 How to Use Generic Prefixing

Terminals come in a variety of forms, so defining them to CA-Top Secret can seem to be an overwhelming task. However, you can make terminal definition easier by using generic prefixing. If a generic prefix is used, it must be from one to eight characters in length. A sample command for assigning ownership of JES remote 19, reader 3 to the Accounting Department appears below.

```
TSS ADD(ACTDEPT) TERM(R19.RD3)
```

The following tables provide instructions and examples on how to specify prefixes for each type of terminal. Generic prefixing is discussed in detail in Chapter 6, *Resource Security Validation*.

For VM: Terminal definitions for VM are shown in the table below.

Table 7-1. Terminal Definitions for VM		
Type	Prefix	Example
Locally attached	GRAF plus four-character local address	TSS ADD(BUDEPT) TERM(GRAF02BA)
Remotely attached VM-controlled network terminals	NETW plus four-character resource id	TSS ADD(CORP) TERM(NETW0301)
Logical devices	LDEV plus four-character address of logical device which is arbitrarily defined.	TSS ADD(CORPNET) TERM(LDEV1234)
VTAM/SNA	8-character LU name	TSS ADD(FINDEPT) TERM(XXXXXXXX)

Note: The four-character address for logical devices is arbitrarily assigned by CP when a product such as VM/PASSTHRU or CA-VTERM requests such a device. LDEV is the only practical prefix when specifying a logical device with TSS ADDTO or PERMIT.

For PCs: Terminal definitions for PCs are shown in the next table.

Table 7-2. Terminal Definitions for PCs		
Type	Prefix	Example
PC	8-character LU name	TSS ADD(DEVDEPT) TERM(XXXXXXXX)

7.5.1.2 How to Authorize Access

The PERMIT command function is used to let designated users access the specified terminals in either an unlimited or restricted manner.

Note: Terminal security is bypassed when the NORESCHK attribute is specified.

Restrictions are specified by incorporating the appropriate PERMIT parameters. For example, the following PERMIT allows USER01 to access terminal K18L1125 from Monday through Friday.

```
TSS PER(USER01) TERM(K18L1125)
    DAYS(WEEKDAYS)
```

You can use the common restrictions (like time and duration) discussed in Chapter 5, *Password and System Entry Security*. However, terminals can't be restricted using access levels or the program pathing option. For a complete list of the parameters that can be used with PERMIT see the *Command Functions Guide*.

7.5.1.3 How to Access All Terminals

To allow a user to access all protected terminals, use the *ALL* indicator. However, before using *ALL* you must assign ownership of it to the MSCA, as shown in the next example.

```
TSS ADD(MSCA) TERM(*ALL*)
```

After ownership has been assigned, you can use the indicator in a PERMIT statement like the one shown below.

```
TSS PER(USER01) TERM(*ALL*)
```

7.6 Protecting Programs

You can protect any program through CA-Top Secret regardless of whether the program is resident, in a private job, or a step library. As a result, sensitive system, proprietary, and applications programs—such as VTOC updates, tape initialization programs, and critical application modules—can be protected from unauthorized use. For example, programs specified by an EXEC PGM MSHP statement and all programs executed internally via LINK, LOAD, XCTL, and ATTACH commands can be protected.

Any owned program is protected. An undefined program isn't protected (unless the DEFPROT attribute has been added to the PROG resource class in the RDT).

Like other ownable resources, programs must first be owned before being authorized. To establish ownership, use a TSS CREATE/ADD PROGRAM entry like the one shown below; then use a TSS PERMIT PROGRAM entry to specify authorizations. The following example protects the program IEHINITT by assigning ownership of it to DEPT01.

```
TSS ADD(DEPT01)  PROG(IEHINITT)
```

How to Remove Ownership: Removing ownership of a program is a two step process.

Step 1 Revoke all permissions for the resource. A sample command appears below.
TSS REV(USER01) PROG(IEHINITT)

Tip: Use TSS WHOHAS to find out who has PERMITs for the resource.

Step 2 Remove the ownership of the program. A sample command appears below.
TSS REMOVE(DEPT01P) PROG(IEHINITT)

Note: CA-Top Secret won't remove ownership unless all permissions are revoked.

7.6.1 How to Assign Default Protection

Default protection gives security protection to a program even if that program hasn't yet been defined to CA-Top Secret. As a result, a security violation will occur if a request is made to access any unowned program.

To give default protection to programs, attach the DEFPROT attribute to the resource class named PROGRAM using the TSS REP(RDT) command function as shown below.

```
TSS REP(RDT) RESCLASS(PROGRAM) ATTR(DEFPROT)
```

7.6.2 How to Use Generic Prefixing

Program ownership can be designated with generic prefixes. Once a prefix is owned, any program beginning with that prefix is protected and must be permitted to other ACIDs. If a generic prefix is used, it must be from one to eight characters in length. The following example assigns ownership of all sensitive IBM utilities to the Systems Department.

```
TSS ADD(SYSDEPT) PROG(IEH)
```

Generic prefixing is discussed in detail in Chapter 6, *Resource Security Validation*.

7.6.3 How to Authorize Access

Use the TSS PERMIT command function to allow designated users to access the indicated programs in either an unlimited or a restricted manner. Restrictions are indicated by incorporating the appropriate PERMIT parameter. For example, the following PERMIT allows USER01 to use the IEHINITT utility from 7:00 a.m. to noon.

```
TSS PER(USER01)  PROG(IEHINITT)  TIMES(07,12)
```

To give everyone the ability to use IEHINITT in the morning, enter:

```
TSS PER(ALL)  PROG(IEHINITT)  TIMES(07,12)
```

You can use the common restrictions (like time and duration) discussed in Chapter 5, *Password and System Entry Security*. For a complete list of the parameters that can be used with PERMIT see the *Command Functions Guide*.

7.6.3.1 How to Assign a Program Path

If you are using program pathing, users who must access a resource through an owned "privileged program" must be given authorization through a TSS PERMIT PROG entry before the PRIVPGM attribute can be used. The statement shown below is a sample entry.

```
TSS PER(USER01)  PROG(APUPDATE)
```

Once the above permission has been granted, program pathing is then implemented with the PRIVPGM parameter. The example shown below authorizes USER01 to read data sets whose highest level qualifiers are SALPAY.MASTER, but only through program APUPDATE when running in the Batch facility.

```
TSS PER(USER01)  DSN(SALPAY.MASTER)
                  PRIVPGM(APUPDATE)  FAC(BATCH)
```

7.6.3.2 How to Access All Programs

To allow users access to all programs, use the resource identifier ***ALL***. Before you can use this identifier, you must assign ownership of it to the MSCA, as shown in the following example.

```
TSS ADD(MSCA) PROG(*ALL*)
```

After ownership has been assigned, you can use the identifier in a PERMIT statement as shown below.

```
TSS PER(USER01) PROG(*ALL*)
```

This example allows USER01 to use any protected program.

7.7 Protecting Applications

Using the APPL keyword, online applications that interface with CA-Top Secret through the MVS System Authorization Facility (SAF)—such as those used with DL/I—can be protected. In addition, you can use this keyword to identify the application name of the LU to which a conversation request can be sent to secure APPC conversations.

Like other ownable resources, applications must first be owned before being authorized. To establish ownership, use a TSS CREATE/ADD APPL entry like the one shown below; then use a TSS PERMIT APPL entry to specify authorizations.

```
TSS ADD(DEPT01) APPL(PAYP)
```

The APPL keyword identifies this resource class in the RDT.

Note: For the DL/I batch message processing regions (BMP) and message processing regions (MPP), you should use its Application Group Name (AGN) as the APPL keyword's operand.

How to Remove Ownership: Removing ownership of an application is a two step process.

Step 1 Revoke all permissions for the resource. A sample command appears below.

```
TSS REV(USER01) APPL(PAYP)
```

Tip: Use TSS WHOHAS to find out who has PERMITs for the resource.

Step 2 Remove the ownership of the application. A sample command appears below.

```
TSS REMOVE(DEPT01P) APPL(PAYP)
```

Note: CA-Top Secret won't remove ownership unless all permissions are revoked.

7.7.1 How to Use Generic Prefixing

Application ownership can be designated using generic prefixes. Once a prefix is owned, any application beginning with that prefix is protected and must be permitted to other ACIDs. The following example gives ownership of applications that begin with the prefix PAY to the ACID PAYDEPT.

```
TSS ADD(PAYDEPT) APPL(PAY)
```

Generic prefixing is discussed in detail in Chapter 6, *Resource Security Validation*.

7.7.2 How to Authorize Access

Use the TSS PERMIT command function to allow designated users to access the indicated applications in either an unlimited or a restricted manner. Restrictions are indicated by incorporating the appropriate PERMIT parameter. For example, the following PERMIT allows USER01 to use the application PAYP through the Batch facility.

```
TSS PER(USER01) APPL(PAYP)
FAC(IMSPROD)
```

You can use the common restrictions (like time and duration) discussed in Chapter 5, *Password and System Entry Security*. However, applications can't be restricted using access levels or the program pathing option. For a complete list of the parameters that can be used with PERMIT see the *Command Functions Guide*.

7.8 Protecting Installation-Defined Resources

Protection can be extended to include installation-defined resources. Installation-defined resources are those resource classes that don't usually trigger the MVS Standard Security Interface to issue a security call. Typical installation-defined resources might include:

- account codes
- job classes
- database fields and contents
- library members

Any installation-defined resource class can be implemented as either an ownable or an unownable resource. The decision should depend on the particular characteristics and requirements of that particular resource type:

- If it is useful to be able to put specific restrictions on the use of this resource type using the PERMIT function, then make it an ownable resource.
- If administrative scope and hierarchy are pertinent to who should be able to authorize access, then it should be made ownable.
- If neither of these considerations apply, then make the resource unownable since it will be easier to administer.

CA-Top Secret provides a group of reserved parameters in the RDT that are used to refer to installation-defined resources:

ABSTRACT	Used with additional site-written code to extend security to other "system" type resources, such as job execution classes.
FIELD	Used to define database fields and to validate access to these fields by user program or transactions.
UR1, UR2	Used to designate owned, non-system, site-defined resources, such as particular account codes. They are implemented and function in the same manner as ABSTRACT and FIELD.
USERx	Used to designate unowned types of resources.

Extending security to installation-defined resources can involve customizing VSE exits, the CA-Top Secret exit, or one of the CA-Top Secret application interfaces. See Chapter 11, *Customizing and Extending Security* for information on exits and interfaces.

Like other ownable resources, installation-defined resources must first be owned before being authorized. To establish ownership, use a TSS CREATE/ADD statement like the ones shown next; then use a TSS PERMIT statement to specify authorizations. The following example contains syntax for assigning ownership of an ABSTRACT, FIELD, and UR2 resource.

```
TSS ADD(CORPORAT) ABSTRACT(AC1)
TSS ADD(DEPTC) FIELD(A100,A200,A300)
TSS ADD(CORPORAT) UR2(ACCT7889,ACCT4567)
```

Note: FIELDS aren't automatically protected by CA-Top Secret after they are owned. An application program must perform the security check by calling CA-Top Secret using FRACHECK or the application high-level language interface.

How to Remove Ownership: Removing ownership of an installation-defined resource is a two step process.

Step 1 Revoke all permissions for the resource. A sample command for each parameter appears below.

```
TSS REV(TECHPROF) ABS(AC1)
TSS REV(PRFCLRK) FIE(A1006,A3002)
TSS REV(CLKPROF) UR2(ACCT7889,ACCT4567)
```

Tip: Use TSS WHOHAS to find out who has PERMITs for the resource.

Step 2 Remove the ownership of the installation-defined resource. A sample command for each parameter appears below.

```
TSS REMOVE(CORPORAT) ABS(AC1)
TSS REMOVE(DEPTC) FIE(A100,A200,A300)
TSS REMOVE(CORPORAT) UR2(ACCT7889,ACCT4567)
```

Note: CA-Top Secret won't remove ownership unless all permissions are revoked.

7.8.1 How to Use Generic Prefixing

Ownership for any installation-defined resource can be designated using generic prefixes. Once a prefix is owned, any installation-defined resource beginning with that prefix is protected and must be permitted to other ACIDs. The following example gives ownership of fields beginning with the prefix A100 to the ACID DEPTC.

```
TSS ADD(DEPTC) FIE(A100)
```

Generic prefixing is discussed in detail in Chapter 6, *Resource Security Validation*.

7.8.2 How to Authorize Access

Use the TSS PERMIT command function to allow designated users to access the indicated installation-defined resources in either an unlimited or a restricted manner. Restrictions are indicated by incorporating the appropriate PERMIT parameter. A sample PERMIT statement for each parameter is shown in the following example.

```
TSS ADD(TECHPROF) ABS(AC1) DAYS(FRIDAY)
TSS PER(PRFCLRK) FIE(A1006,A3002) TIMES(08,17) DAYS(WEEKENDS)
    PRIVPGM(ISPTASK1) ACCESS(UPDATE)
TSS PER(CLKPROF) UR2(ACCT7889,ACCT4567) ACCESS(ALL)
```

You can use the common restrictions (like time and duration) discussed in Chapter 5, *Password and System Entry Security* for all of the parameters. However, access levels can only be specified for FIELD, UR1, and UR2. For a complete list of the parameters that can be used with PERMIT see the *Command Functions Guide*.

7.8.2.1 How to Assign Access Levels

Access levels are assigned using the ACCESS parameter. These access levels are defined, but must be checked by the exit or application program. This parameter is used to restrict how FIELDs, UR1, and UR2 resources can be accessed. The following list shows the access levels that can be specified; READ is the default.

Level	Meaning
ALL	Resource can be accessed in any way.
UPDATE	Resource can be updated; READ and WRITE access are implied.
READ	Resource can be read; the default.
WRITE	Data can only be written into the resource.
NONE	Resource can't be used in any way.

The next example authorizes any user in the CLKPROF profile to update the UR2 resources ACCT7889 and ACCT4567.

```
TSS PER(CLKPROF) UR2(ACCT7889,ACCT4567) ACCESS(UPDATE)
```


7.8.2.2 How to Assign a Program Path

If you are using program pathing and a user must have further access to a resource through an owned "privileged program", then access must also be given to the owned program through a TSS PERMIT PROG entry before the PRIVPGM attribute can be used. The statement shown below is a sample entry.

```
TSS PER(PRFCLRK) PROG(IMSPZAP)
```

Once the above permission has been granted, program pathing is then implemented with the PRIVPGM parameter. The example shown below allows any user attached to the PRFCLRK profile to update the specified fields from 8:00 a.m. until 5:59 p.m. on Monday through Friday, but only through the program ISPTASK1.

```
TSS PER(PRFCLRK) FIE(A1006,A3002) TIMES(08,17) DAYS(WEEKENDS)  
PRIVPGM(IMSPZAP) ACCESS(UPDATE)
```

7.9 Protecting Online Transactions

CA-Top Secret secures online transactions in two ways:

1. Through OTRAN (resource) security.

The OTRAN resource class allows transactions—which are ordinarily considered unowned—to be administered as owned resources. By allowing transactions to be owned by various zones, divisions, or departments, security can be administered on a decentralized, application-by-application basis, and access to each transaction can be tailored with the same flexible restriction options available for other resources—such as day, time, access level, and so on.

2. By the Limited Command Facility (LCF).

Through LCF, you can assign authorizations and restrictions to transactions on a user-by-user basis without impacting other users. However, since transactions controlled with LCF are **not** owned, if factors such as administrative scope and resource auditing capabilities are considered highly important for your security environment, an LCF-based approach to transaction security may not be the best approach for you.

Specific details on implementing LCF and OTRAN security appear in each facility's *Implementation* guide. General procedures that apply to all facilities are covered in this section.

7.9.1 Implementing OTRAN Security

The OTRAN resource name is shared by all CICS, CA-IDMS, and DL/1 facilities. Therefore, protecting a transaction using OTRAN for a CICS region also results in transactions of the same name being protected in all CICS, CA-IDMS, and DL/1 regions that are also under the control of CA-Top Secret. As a result, when securing transactions with OTRAN, keep these points in mind:

- OTRAN protects the transaction IDs across **all** facilities.
- Once a transaction is owned, it is protected from use by any user in any facility. As a result, you should consider all the users who may need that transaction (in both test and production regions) before using the OTRAN resource class.
- Transaction security through the OTRAN resource class overrides LCF protection. Once a transaction is owned, any appearance of it in an LCF list, whether inclusive or exclusive, is ignored.
- A transaction protected via OTRAN won't go through LCF checking.
- All features of owned resources are supported by OTRAN (such as TSS WHOHAS and TSS WHOOWNS).
- Password reverification can't be used.

- Screen Level Protection (SLP) can be enabled by associated a MAP record with an OTRAN or PPT resource.

7.9.1.1 How to Set Up OTRAN Security

Like other ownable resources, transactions must first be owned before being authorized. To establish ownership, use a TSS CREATE/ADD OTRAN entry like the one shown below.

```
TSS ADD(acid) OTRAN(transaction)
```

Using the syntax shown above, if you want to have the Payroll Department (whose ACID is PAYDEPT) own the transaction PAYR, enter:

```
TSS ADD(PAYDEPT) OTRAN(PAYR)
```

Once ownership is established, use a TSS PERMIT OTRAN entry to specify authorizations. Instructions for authorizing access appear in the next section.

How to Remove Ownership: Removing ownership of a transaction is a two step process.

Step 1 Revoke all permissions for the resource. A sample command appears below.
TSS REV(PAYPROG) OTRAN(PAYR)

Tip: Use TSS WHOHAS to find out who has PERMITs for the resource.

Step 2 Remove the ownership of the transaction. A sample command appears below.
TSS REMOVE(PAYDEPT) OTRAN(PAYR)

Note: CA-Top Secret won't remove ownership unless all permissions are revoked.

7.9.2 How to Use Generic Prefixing

OTRAN resources can be designated using generic prefixes. Once a prefix is owned, any transaction beginning with that prefix is protected and must be permitted to other ACIDs. The following example gives ownership of transactions that begin with the prefix PAY to the ACID PAYDEPT.

```
TSS ADD(PAYDEPT) OTRAN(PAY)
```

Generic prefixing is discussed in detail in Chapter 6, *Resource Security Validation*.

7.9.2.1 How to Authorize Access

The PERMIT command function is used to allow designated users to use the specified transactions in either an unlimited or restricted manner. Restrictions are specified by incorporating the appropriate PERMIT parameters. The following example allows a user whose ACID is PAYPROG to access the transaction PAYR.

```
TSS PER(PAYPROG) OTRAN(PAYR)
```

Commonly used restrictions (like time, duration, and facility) are discussed in Chapter 5, *Password and System Entry Security*. For a complete list of the parameters that can be used with PERMIT see the *Command Functions Guide*.

7.9.2.2 How to Assign a Program Path

Program pathing can be used to restrict the access of certain transactions to designated programs. The following example allows PAYPROG to start the transaction PAYR through the PAYGRP program.

```
TSS PER(PAYPROG) OTRAN(PAYR) ACC(EXECUTE)
PRIVPGM(PAYGRP)
```

Note: Program pathing isn't supported by all facilities.

Program pathing is discussed in detail in Chapter 6, *Resource Security Validation*.

7.9.3 Implementing LCF Security

If you choose not to protect transactions using OTRAN, you can extend protection to them using the Limited Command Facility (LCF). Resources protected by LCF are known as unownable resources. LCF resources are added to individual ACIDs to provide access. They are not permitted. LCF security allows the security administrator to limit:

- Commands available to a TSO user.
- Monitor commands available to CA-Roscoe, ACEP, WYLBUR, and other online multi-user applications.
- CICS, DL/1, and CA-IDMS transactions.
- Programs a batch job can execute.
- ISPF/SPF panel options available to a user.

When securing transactions with LCF, keep these points in mind:

- LCF transaction security is easy to implement and is the most common method of implementing security.
- Implementation can be performed for one user without impacting others.
- LCF allows you to protect transactions without interfering with other facilities. (You can set up LCF security on your test system without affecting your production system.)
- Unownable resources must still be defined by class to the RDT, but individual resource access restrictions are administered through the Limited Command Facility (LCF).
- LCF level security shouldn't be combined with OTRAN (resource) security. If a transaction is secured by both LCF and OTRAN, OTRAN overrides LCF.

- Password reverification can be used for transactions that are protected through LCF. Reverification isn't available for transactions protected by OTRAN.
- The XDEF and NOXDEF FACILITY suboptions are used for LCF to provide default protection.
- The LCFTRAN and LCFCMD FACILITY suboptions are used to tailor the text for CA-Top Secret LCF messages.
- The TSS WHOHAS command isn't valid with LCF security. (It is available for OTRAN.)

It is recommended that within a single facility either inclusive or exclusive LCF be used to secure sensitive transactions (**not** a combination of both). If both types of LCFs are used within a facility, inappropriate authorizations or failures of requests may be made due to the CA-Top Secret decision-making process.

7.9.3.1 How to Set Up LCF Security

Transactions protected through LCF must be defined by facility. It is recommended that transactions be divided by function or subset and defined as a group within profiles—this way transactions are defined only once per group, instead of once per user. You can then limit access to the transaction with either an inclusive list that specifies which transactions the user is allowed to use, or an exclusive list that specifies which transactions the user isn't allowed to use.

How to Limit Access With Inclusive Lists: The inclusive approach restricts an ACID to using **only** specifically authorized transactions for a particular facility. An inclusive list is indicated by the CMDS or TRANS keyword. (These keywords are interchangeable.) When specifying an inclusive list, use the syntax shown below.

```
TSS ADD(acid) TRANS(facility,(trans,trans...))
```

The following example restricts the users connected to the PROF01 profile to using only the DDYA, DDYU, and DDYY transactions under the CICSPROD facility.

```
TSS ADD(PROF01) TRANS(CICSPROD,(DDYA,DDYU,DDYY))
```

How to Limit Access With Exclusive Lists: The exclusive approach allows an ACID to use all transactions **except** a specific list of restricted transactions for a particular facility. An exclusive list is indicated by the XCMDS or XTRANS keyword. (These keywords are interchangeable.) When specifying an exclusive list, use the syntax shown below.

```
TSS ADD(acid) XCMDS(facility,(trans,trans...))
```

The following example allows USER01 to use all of the TSO transactions except the SPF EDIT (SPF2) panel.

```
TSS ADD(USER01) XCMS(TSO,(SPF2))
```

The LCF Decision Process: When CA-Top Secret is validating an ACID's request to access a command, the following decision-making process is used:

- CA-Top Secret searches the user's Security Records in the following order:
 1. The ACID's Security Record
 2. Any PROFILE(s) associated with the ACID (in the order shown by a TSS LIST command function)
 3. The ALL Record
- The search continues through all records (user, profile, ALL) and ends when an exact match or prefix is found, or when all records are searched. If CA-Top Secret doesn't find an exact or closest match to a transaction and only an inclusive list is encountered during the search of the user's applicable records, then the transaction must be outside the user's authorized list and the transaction is rejected.
- If CA-Top Secret doesn't find an exact or closest match and only an exclusive list is encountered during the search of the user's applicable records, then the transaction must be outside the user's unauthorized (exempt) list and the transaction is accepted.

If Both Inclusive and Exclusive LCF Are Used: As stated previously, using both inclusive and exclusive LCF lists within a single facility is **not** recommended. A mixture of LCF-types is very difficult for any CA-Top Secret security administrator to maintain, whether they are in the same or different security records. However, there are valid circumstances for a mixture in which inclusive LCF is defined for specific users, while exclusive LCF appear in the ALL Record.

If you have decided to use both inclusive and exclusive lists, make sure your definitions don't contradict each other, as the ones shown below do.

```
TSS ADD(CCC) TRANS(CICSA,(ABCT))
TSS ADD(CCC) XTRANS(CICSA,(ABCT))
```

CA-Top Secret uses the following decision-making process if the list types are mixed:

- The order in which an ACID's security records are checked is the same as the order listed previously.
- Inclusive LCFs are checked first; if a match is found, the request is allowed.
- If the transaction isn't found on an inclusive LCF, the exclusive LCF is checked. If the transaction is found on the exclusive LCF, the request is failed.

Note: Within each level of the ACID's Security Records, inclusive then exclusive checking is performed before the next record level, (for example Profile) is reached.

After all levels of checking have been completed, and if both exclusive and inclusive LCF were executed during the security record search, then inclusive LCF logic takes precedence and the record is rejected.

7.9.3.2 How to Provide Default Protection

You can provide default protection for transactions by using the XDEF and NOXDEF suboptions of the FACILITY control option.

- The NOXDEF suboption is set **by default** to allow all users to execute any transaction until access controls have been established by either an inclusive or exclusive list for the user.
- The XDEF suboption indicates that users must have some kind of transaction list—either inclusive (TRANS or CMD) or exclusive (XTRANS or XCMD)—before the user can execute any LCF transaction.

The following example shows how to specify the XDEF option for the CICSPROD facility.

```
TSS MODIFY(FAC(CICSPROD=XDEF))
```

Note: In FAIL mode, a transaction can't be executed by a user unless that user is defined to CA-Top Secret.

7.9.3.3 How to Bypass LCF Security

To specify that a particular user or profile is to bypass all LCF security, attach the NOLCFCHK bypass attribute to its ACID. NOLCFCHK is a powerful attribute; therefore, its use should be carefully restricted. The following example allows the ACID SUPRACID to bypass all LCF security checking.

```
TSS ADD(SUPRACID) NOLCFCHK
```


7.9.3.4 How to Use Generic Prefixing

You can use generic prefixes when specifying transactions within an inclusive or an exclusive list. When using generic prefixing, you must append a G to the prefix, as shown in the next example, to indicate that a generic prefix is being supplied.

```
TSS ADD(PDGRLP) XTRANS(BAT, (IEH(G)))
```

This example restricts the ACID PDGRLP from using all batch programs that begin with the characters IEH.

7.9.3.5 How to Use Reverification

To reduce the chance of someone taking advantage of an unattended terminal, use reverification to force the terminal's user to supply his password to execute a particular transaction. When using reverification, you must append a V to the transaction as shown in the next example.

```
TSS ADD(USR01) TRANS(CICSPROD, (PAY9(V)))
```

For details on how to specify passwords, see Chapter 5, *Password and System Entry Security*.

Note: Reverification is only supported for CA-IDMS 10.2, CICS, and DL/I.

7.9.3.6 How to Access All Transactions

You can use an exclusive LCF list to permit access to all transactions in a facility that, by default, protects all transactions. To allow access to all transactions within a facility, specify an exclusive LCF list for a transaction that doesn't exist. The following example specifies an exclusive LCF list for the CICS facility, listing NULL as the restricted transaction. Because NULL doesn't exist, USER01 is allowed access to all CICS transactions.

```
TSS ADD(USER01) XTRANS(CICS, (NULL))
```

7.9.4 Implementing Screen Level Protection (SLP)

This section explains how to implement Screen Level Protection (SLP). SLP gives you detailed control over which users have access to the data displayed on a screen. This access is controlled by defining which fields on a screen you want to protect to a reserved ACID called the Static Data Table (SDT) record, and then permitting access to the fields using the PERMIT command.

The SDT contains two record elements that are used to implement SLP. They are:

- MAPREC** Defines the layout of a CICS map, including field name, row, column, and length.
- SELECT** Defines the logic, using Boolean expressions, that specifies who can view and/or change the screen's fields.

Implementing SLP is a four step process:

- Step 1** Gather Information
- Step 2** Enter Definitions
- Step 3** Permit Access to the Defined Maps
- Step 4** Enable Protection

Each step is described in detail in the following sections.

7.9.4.1 Gather Information

Before you can define record elements, there are several preliminary steps you must perform. These steps are important, since the information you gather here will determine how smoothly SLP is implemented.

- Step 1** Determine which of your applications would benefit from SLP.
- Step 2** Meet with the programmers to gather information about the application (like field names, data types, length of field, and selection criteria).
- Step 3** Become familiar with the application.
- Step 4** Plan the details needed to implement SLP for this application. For example, you may decide on a selection criteria that limits who can view salary information.
- Step 5** Determine who will be the administrator(s) for implementing SLP and give them MISC3(SDT) authority.

7.9.4.2 Enter Definitions

Once you have completed the preliminary steps as outlined in the previous section, you are ready to begin entering the definitions into the SDT. All definitions are entered using the TSS ADD(SDT) command.

Note: Depending on the number of field map you are protecting, these steps can be labor intensive.

- Step 1** Define the MAPREC definitions to the SDT. A sample definition is shown below.

```
TSS ADD(SDT) MAPREC(msdept) MAPDATA(mdept,10,8,4)
```

Note: If you are protecting multiple field maps within one screen, you must do a separate ADD for each one you want to protect.

- Step 2** Define the SELECT expressions to the SDT that you will be using on the PERMIT command. A sample definition is shown below.
- ```
TSS ADD(SDT) SELECT(dp1000)
 SELDATA('IF dept EQ "" OR dept GE "1000" AND dept LE "1099")
```
- Note:** You need to check if the field is null so that Screen Level Protection will allow the transaction to continue, if it finds no data (null) within the terminal screen.
- Step 3** Check your work by listing the SDT records you just created. To list all records, use the command shown below.
- ```
TSS LIST(SDT) RECORD(ALL)
```
- Step 4** Use TSS REPLACE(SDT) to correct any errors.
- Step 5** When you are satisfied that everything is correct, refresh the SDT in-core tables using the command shown below.
- ```
TSS MODIFY(SDTTABLE)
```

Refer to Section 8.3, Static Data Table (SDT) Record, for a detailed description of the keywords used with TSS ADD(SDT).

### 7.9.4.3 Permit Access to the Defined Maps

When your definitions are complete, you are ready to permit access to the defined maps.

1. First, you must revoke any existing PERMITs that a user may have for this OTRAN or PPT resource.
2. Then, re-PERMIT the resources using the SELECT and MAPREC clauses. A sample PERMIT command is shown below.

```
TSS PERMIT(jane) OTRAN(PAYR) ACCESS(ALL) SELECT(dp1000) MAPREC(ENG1)
```

### 7.9.4.4 Enable Protection

After your definitions and permissions are complete, you must enable SLP for the facility. (The definitions and permissions will not take effect until SLP is enabled.) Use the command shown below to enable SLP in the CICS region.

```
TSS MODIFY FAC(cicsprod=SLP=YES)
```



## Chapter 8. Maintaining Special Security Records

---

CA-Top Secret usually has a user ACID, or a similar unique ACID name belonging to organizations, profiles, or Control ACIDs, as the target of the TSS command. However, there is a group of special or reserved ACIDs, predefined by CA-Top Secret, whose names are the specific target of the TSS command. These special ACIDs include:

|                                               |                                                                                                               |
|-----------------------------------------------|---------------------------------------------------------------------------------------------------------------|
| <b>RDT (Resource Descriptor Table) Record</b> | Stores both predefined and user-defined resources.                                                            |
| <b>FDT (Field Descriptor Table) Record</b>    | Stores both predefined and user-defined fields.                                                               |
| <b>SDT (Static Data Table) Record</b>         | Stores internal, non-volatile data used to protect records, fields, screens, calendars, and other resources.  |
| <b>NDT (Node Descriptor Table) Record</b>     | Contains data for assigning Pass Tickets and Session Keys to applications. It also contains VAX-related data. |
| <b>ALL Record</b>                             | Identifies resources that are globally accessible to all users.                                               |
| <b>STC (Started Task Command) Record</b>      | Defines an MVS started task command to CA-Top Secret.                                                         |
| <b>AUDIT Record</b>                           | Audits a specific resource.                                                                                   |
| <b>DLF (Data Lookaside Facility) Record</b>   | Defines those data sets that are eligible for the IBM Data Lookaside Facility.                                |

Although they aren't the target ACID of their TSS command function, SMS fields are also included in this chapter.

## 8.1 Resource Descriptor Table (RDT) Record

The Resource Descriptor Table (RDT) is a special ACID that defines resource classes and their properties. The RDT contains predefined CA-Top Secret resource classes, including resources used and reserved by other Computer Associates product interfaces. In addition, the RDT can contain user-defined resource classes.

For managing the contents of the RDT Record, you can specify attributes, access levels, and default access levels through the TSS command. This section covers the following topic:

- Modifying the RDT
  - reasons for modifying existing resource classes
  - reasons for defining new resources classes
- Maintaining the RDT
  - necessary administrative authority
  - how to define and list the RDT
  - how to change and remove values in the RDT

### 8.1.1 Modifying the RDT

You can have one or more reasons for updating or modifying the RDT. These reasons can be divided into two main categories:

#### 1. Modifying Existing Resource Classes

You want to modify an existing resource class to change its attributes. For example, you can add default protection by assigning the DEFPROT attribute to the appropriate TSS command function. DEFPROT and the other attributes are discussed later in this chapter.

**Note:** For predefined resource classes only the EXIT, DEFPROT, MERGE, ALLMERGE, MASK, and GENERIC attributes can be modified.

#### 2. Defining New Resource Classes

You want to define non-CA-Top Secret resource classes in the following situations:

- IBM products that have requirements for a non-fixed or installation-defined resource class name.

Under normal circumstances CA-Top Secret will have anticipated the use of standard IBM resource classes; however, there are exceptions (for example, the resource class used by OPC/A for its resources).

- OEM software packages have unique resource class names that aren't defined by CA-Top Secret. While many security interfaces are predefined to CA-Top Secret, there are exceptions where you must supply the resource class names.

- You must also define resource class names for installation-written security with its own resource classes, and any other customized site applications with resources and security of its own. For example, a site-written application may require the definition of a resource class of \$PROC to provide JCL procedure security as checked by an installation-written JES exit.

**Note:** Installation-written security may require separate naming conventions and access levels.

## 8.1.2 Maintaining the RDT

The TSS commands that relate to the RDT appear in the following sections. For complete information, refer to the *Command Functions Guide*

**Administrative Authority:** You must have MISC1(RDT) authority to ADD, REMOVE, or REPLACE resources in the RDT.

### 8.1.2.1 Defining a Resource to the RDT

To define a new resource class to the RDT, enter the TSS ADD command functions shown below.

```
TSS ADD(RDT) RESCLASS(resource-class-name) RESCODE(hex-code) MAXLEN(nnn)
 [ATTR(attribute-list)] [ACLST(access-level-list)]
 [DEFACC(default-access-level)]
```

The following two keywords are required when adding a new resource to the RDT:

**RESCLASS** Defines an up to or eight-character *resource-class-name* that allows one *resource-class-name* per command; the name can contain letters, numbers, or special characters. The TSS command, logging, and the security interface honor this name. **To avoid any possibility of a user-defined resource conflicting with any future CA-Top Secret predefined resource class, it is recommended that the user-defined resource class have a national character (@, #, \$) or number (0-9) in one of the first four characters of the name.**

**Note:** The beginning characters of a new RESCLASS name can't match the beginning characters of an existing RESCLASS name. For example, VOLUME is an existing predefined RESCLASS name; therefore, it isn't possible to create a new RESCLASS name called VOL.

**RESCODE** Adds the two-digit hexadecimal code ranging between 01 and 3F that allows one *hex-code* per command. RESCODE is used internally by CA-Top Secret to identify the type of resource passed to FRACHECK. This identifier traces and logs information.

**MAXLEN** Defines the maximum length for the user resource class being created.

Optional parameters that can be used when defining a resource to the RDT Record or when modifying an existing resource class follow.

**ATTR** Defines one or more of the following operands:

**EXIT** Calls the installation exit for this resource class.

**NOEXIT** Deactivates the installation exit.

**DEFPROT** Protects this resource class by default.

**NODEFPROT** Deactivates default protection.

**GENERIC** Supports generic prefixing for this resource class.

**NONGENERIC** Deactivates generic prefixing and treats the resource names as fully qualified names.

**PRIVPGM** Supports privileged program for this resource class.

**NOPRIVPGM** Deactivates privileged program support.

**LIB** Supports LIB with PRIVPGM for this resource class.

**NOLIB** Deactivates support for a library with privileged program.

**LONG** Supports 44-character permissions.

**SHORT** Supports eight-character resources.

**MASK** Supports masking characters in a TSS PERMIT.

**NOMASK** Turns off previously changed MASK attribute.

**MERGE** Overrides the AUTH control option and uses the MERGE algorithm to determine the processing record for security validation of this RESCLASS. For a description of the algorithm, refer to Chapter 7, *Resource Security Validation*.

**NOMERGE** Removes the MERGE attribute from the resource.

**ALLMERGE** Overrides the AUTH control option and uses the MERGE algorithm to determine the processing record for security validation of this RESCLASS. For a description of the algorithm, refer to Chapter 7, *Resource Security Validation*.

**NOALLMERGE** Removes the ALLMERGE attribute from the resource.

**VMUSER** Supports VMUSER for this resource class.

**NOVMUSER** Deactivates VMUSER support.



**MASK** Supports masking for this resource class.

**NOMASK** Deactivates masking for this resource class.

**Note:** For PRIVPGM, LIB, and VMUSER the security driver must also support these features. For all predefined CA-Top Secret resource classes (such as DATASET and VOLUME), only the DEFPROT, EXIT, MERGE, ALLMERGE, and GENERIC attributes can be altered through the TSS REPLACE(RDT) command function.

If you defined a resource as maskable, all those currently signed on users with that permitted resource type will start to fail. Those users will have to refresh their security environments. The change to the RDT not only sets up for the new permissions but causes all of the security validations to treat that class as maskable. Those that logged on before the switch have the wrong internal record format and are required to refresh or log off and log on.

For example, to add #PRODUCT to the RDT Record, and give it default protection, enter the command:

```
TSS ADD(RDT) RESCLASS(#PRODUCT) RESCODE(10)
 ATTR(DEFPROT)
```

To remove an attribute that was assigned to a specific resource class, simply use the TSS REPL(RDT) command function and prefix the attribute with NO as explained in 8.1.2.2, "Changing Values in the RDT" on page 8-7.

### **ACLST**

Adds up to 20 access levels for this resource class. If not specified, the resource class does not support access level checking. It is recommended that ALL, CONTROL, UPDATE, and READ be defined.

If the predefined access levels are used, you can simply specify the access level list shown below.

```
ACLST(READ,WRITE)
```

However, if you want your own unique access levels, you must specify the hexadecimal values associated with each access level as illustrated in the following example.

```
ACLST (XYZ=0600,ABC=0005)
```

You can also mix defined access levels with your own unique access levels shown below.

```
ACLST(XYZ=0600,READ)
```

The access level list is supported both by the TSS command during administration and access validation, and for logging and reporting. CA-Top Secret predefined access levels are listed below with their hexadecimal values:

```
NOCREATE(0100)
```

NONE(0000)  
 PURGE(0100)  
 BROWSE(0200)  
 FEOV(0200)  
 CONTROL(0400)  
 MULTI(0400)  
 REPL(0800)  
 SCRTCH(0800)  
 CREATE(1000)  
 DELETE(1000)  
 FIND(1000)  
 NONSHR(2000)  
 WRITE(2000)  
 MWRITE(2400)  
 READ(4000)  
 SHR(4000)  
 MREAD(4400)  
 BLP(8000)  
 FETCH(8000)  
 UPDATE(8000)  
 LOGON(8000)  
 ALL(FFFF)

To remove an access level list, refer to 8.1.2.2, “Changing Values in the RDT” on page 8-7.

**DEFACC**

Sets the default access level CA-Top Secret assigns on a TSS PERMIT. If not specified, the default access is NONE. The predefined CA-Top Secret access levels with their respective hexadecimal values are listed above.

**Note:** The access level specified by DEFACC must match the applicable access levels indicated by the ACLST entries for that resource. If they don't match; if no ACLST was specified, you will receive a TSS0282E error message.

When creating a user-defined resource class, remember the following rule: If the access level is not one that is known to CA-Top Secret, you must specify the hexadecimal value in the DEFACC as well as the ACLST field shown below.

```
TSS ADD(RDT) RESCLASS($NEWRES) RESCODE(04)
 ACLST(ALLOW=4000) DEFACC(ALLOW=4000)
```

**Examples:** To add a new resource class to the RDT Record with READ and WRITE access levels, enter the command shown below.

```
TSS ADD(RDT) RESCL($PAY) RESCODE(12) ACLST(WRITE,READ)
```

To give appropriate administrative authority for ownership and granting users access to a new resource class \$PAY, use the command:

```
TSS ADMIN(ADM01) $PAY(OWN,XAUTH) ACC(READ,WRITE)
```

You can now add this new resource class name \$PAY to a department that is within your scope by entering:

```
TSS ADD(DEPT01) $PAY(401K)
```

**Note:** 401K is a resource name that is now accessed by the resource class \$PAY.

To permit this resource to USER01 with READ access, enter the TSS command shown below.

```
TSS PER(USER01) $PAY(401K) ACC(READ)
```

### 8.1.2.2 Changing Values in the RDT

To change the values of previously-defined resource classes, enter the command shown below.

```
TSS REPLACE(RDT) RESCLASS(resource-class-name)
 ATTR(attribute-list) ACLST(access-level-list)
 DEFACC(default-access-level)
```

**RESCLASS** Contains an up to eight-character *resource-class-name* whose values are changed or removed; one *resource-class-name* is allowed per command, and the name can contain letters, numbers, or special characters. You **must** use at least one of the keywords listed below with RESCLASS when changing a value of an existing resource in the RDT.

**ATTR** Removes an attribute from the *attribute-list* that was assigned to a specific resource class, using the TSS REPL(RDT) command function and prefixing the attribute with NO to remove it. For example, the resource class #PRODUCT no longer has default protection by entering the command shown below.

```
TSS REPLACE(RDT) RESCLASS(#PRODUCT) ATTR(NODEFPROT)
```

To remove the LONG attribute that is already attached to a specific user-defined resource class, specify ATTR(SHORT). This is the **only**

exception that does not use the prefix NO when attempting to remove an existing attribute.

**Note:** For all predefined CA-Top Secret resource classes, only the DEFPROT, EXIT, MERGE, ALLMERGE, and GENERIC attributes can be altered through the TSS REPLACE(RDT) command function.

**ACLST**

Changes an access level from the *access-level-list* of the existing resource.

To remove and replace an access level from the *access-level-list*, use the following TSS REPLACE command function which, in the following example, removes an existing READ access and replaces it with UPDATE access.

```
TSS REPLACE(RDT) RESCLASS(#PRODUCT) ACLST(UPDATE)
```

If you want to keep the READ access level and add the UPDATE level, enter the command shown below.

```
TSS REPLACE(RDT) RESCLASS(#PRODUCT) ACLST(READ,UPDATE)
```

To remove an access level entirely from the *access-level-list*, enter the TSS REPLACE command function and the ACLST keyword followed by parentheses shown below.

```
TSS REPLACE(RDT) RESCLASS(#PRODUCT) ACLST()
```

**DEFACC**

Changes a *default-access-level* assigned to an existing resource.

To change a default access level, use the TSS REPLACE command function shown below, which removes the existing READ access and replaces it with NONE.

```
TSS REPLACE(RDT) RESCLASS(#PRODUCT) DEFACC(NONE)
```

### 8.1.2.3 Removing a Resource in the RDT

To remove a user-defined resource class definition, enter the command shown below.

```
TSS REMOVE(RDT) RESCLASS(resource-class-name)
```

**RESCLASS**

Contains an up to eight-character *resource-class-name* that allows one *resource-class-name* per command, and is the only keyword that is required.

For example, if you want to remove #PRODUCT from the RDT, enter the command below.

```
TSS REMOVE(RDT) RESCLASS(#PRODUCT)
```

**Note:** Removing a resource class from the RDT requires that all owned resources belonging to that particular resource class were previously removed.

### 8.1.2.4 Listing the RDT

**Listing the Entire Record:** To list the entire RDT, enter the command shown below.

```
TSS LIST(RDT)
```

**Listing a Resource Class:** To list a resource class, enter the command shown below.

```
TSS LIST(RDT) RESCLASS(resource-class-name)
```

**RESCLASS** Lists an up to eight-character *resource-class-name* that allows one *resource-class-name* per command, and is the only keyword that is required.

For example, to list data relating to the #PRODUCT resource class, enter the command shown below.

```
TSS LIST(RDT) RESCLASS(#PRODUCT)
```

You can limit the authority of an ACID to only listing the RDT, by assigning MISC8(LISTRDT) authority.

### 8.1.3 Resource Masking

With Release 3.0, the resource masking feature enables you to perform the following:

- Change existing non-maskable resources to support masking by simply entering the following command

```
TSS REPLACE(RDT) RESCLASS(resclass) ATTR(MASK)
```

which will convert the resource to support masking although the maximum number will remain at eight.

If you have an existing resource that you want to own which is longer than eight characters, you must revoke and remove all users of that resource class, remove the resclass from the RDT, and re-create the resource class using the method described in the next item.

- Create user-defined resources to support ownership longer than eight characters by specifying the following:

## 8.1 Resource Descriptor Table (RDT) Record

TSS ADD(RDT) RESCLASS(resclass) ATTR(MASK)

- List the resource class to show several new features such as: a MASKABLE or NOMASK resource, a MAXOWN(nn) attribute displaying the maximum value for an ownable resource, and a MAXPERMIT(nnn) attribute displaying the maximum value for a permitted resource.
- Define lengths up to 255 characters with the MAXLEN(nnn) keyword. In prior releases, ATTR(LONG) implies MAXLEN(44).

For more information on resource masking, refer to Chapter 2 of the *Command Functions Guide*.

## 8.2 Field Descriptor Table (FDT) Record

The Field Descriptor Table is a reserved or special ACID that contains and manipulates fields. These fields give you the ability to define variable length data to the Security File, and can be predefined by Computer Associates or user-defined. Before attaching a field to an ACID, it must first be defined to the FDT. Predefined fields can be modified immediately since they already exist. A sample of some of the predefined fields you can find in the FDT appear below.

|          |         |
|----------|---------|
| LANGUAGE | SMSAPPL |
| PCADMIN  | OPIDENT |
| LTIME    | SMSDATA |
| PCDSDAYS | OPPTY   |
| OPCLASS  | SMSGMT  |

## 8.2.1 Modifying the FDT

You can have one or more reasons for updating or modifying the FDT. These reasons can be divided into two main categories:

### 1. Modifying Existing Fields

You want to modify an existing field to change its characteristics. For example, if you want to change the displayed title of a field shown when listing a user, you would enter:

```
TSS REP(FDT) FDTNAME(FIELD1) DISPLAY('USER FIELD1')
```

### 2. Defining New Fields

- You want to use fields to define installation information (by user ACID) that can be manipulated or extracted by application programs. Applications or the Application Interface can extract and update this information using the RACROUTE macro.
- Once defined, you can also add these fields to ACIDs using the TSS commands. You can ADD, REMOVE, REPLACE, or LIST the defined data the same way you would with fields predefined by CA-Top Secret.

**Note:** The Security File is structured to provide CA-Top Secret with the best performance possible. However, a wide use of user-defined data can cause a situation where the original CA-Top Secret information becomes a fraction of the total Security File—causing an unbalanced I/O as well as a misuse of storage and CPU. You should re-examine the Security File size according to the amount of user data that the file is expected to hold.

## 8.2.2 Maintaining the FDT

The TSS commands that relate to the FDT appear in the following sections. For complete information, refer to the *Command Functions Guide*.

**Administrative Authority:** You must have MISC1(RDT) authority to ADD, REMOVE, or REPLACE fields in the FDT Record.



### 8.2.2.1 Defining New Fields to the FDT

You can define a new field to the FDT by using the TSS ADD(FDT) command function shown below.

```
TSS ADD(FDT) FDTNAME(field-name) FDTCODE(hex-code)
 SEGMENT(segment-name) MAXLEN(nn) DISPLAY(display-name)
 [ATTR(MIXED)]
 [(NONDISP)]
```

**FDTNAME** Adds an up to eight-character user-defined field to the FDT Record when FDT is the target ACID name, and allows one *field-name* per command that can be a letter, number, or special character.

**FDTCODE** Adds a user-defined *hex-code* to the FDT Record when FDT is the target name, and allows one *hex-code* per command that can range from 01 to FF.

**SEGMENT** Assigns an up to eight-character field to a specific segment, and allows one *segment-name* per command that can be a letter, number, or special character. You can't add user-defined fields to predefined segments. The following are predefined CA-Top Secret segments:

|          |         |          |
|----------|---------|----------|
| ALL      | CA-PC   | DFP      |
| DL/1     | OMVS    | TSO      |
| BASE     | CICS    | DLFDATA  |
| LANGUAGE | OPERMAN | WORKATTR |

A new segment is defined by specifying a unique value when adding a new field. If the value specified for a segment already exists, a new field is added to that segment. If the segment name is not found on the file, a new segment is created.

**MAXLEN** Defines the number of bytes that can be entered for the user-defined FDT entries, and allows one MAXLEN value per command. The total of all user-defined fields should not exceed 32,767 bytes.

**DISPLAY** Defines an up to 11-character *display-name* with its associated defined field and segment in the FDT Record, and allows one *display-name* per command that can be a letter, number, or special character. The *display-name* must be enclosed in single quotes if it contains blanks.

**ATTR** Displays fields from the FDT in mixed case format when using the MIXED attribute. If you do not specify ATTR, it defaults to upper-case.

If a field is defined with the NONDISP attribute, the field cannot be seen by an administrator using a TSS LIST function on the ACID. The data can be extracted and/or modified as normal using the Application Interface or the RACROUTE macro.

**Note:** You may want to use non-CA-Top Secret fields to define installation information (by user ACID) that can be maintained or extracted by application programs. Applications can extract and update this user information using the RACROUTE macro. (For syntax of the RACROUTE macro, see any IBM manual that explains the external security interface.)

**Examples:** This example creates an area code and home phone field called HPHONE that will belong to a segment named HPHNATTR and a display called HPHNUM. The maximum length for the field is 12 bytes and has a hex-code of 01. This field is created by entering the command shown below.

```
TSS ADD(FDT) FDTNAME(HPHONE) FDTCODE(01)
 SEGMENT(HPHNATTR) MAXLEN(12) DISPLAY(HPHNUM)
```

To add the contents of this area code and home phone field to its respective owner, enter the command shown below.

```
TSS ADD(USER01) HPHONE('908 780 5550')
```

**Note:** Use single quotes when the field contains blanks.

### 8.2.2.2 Changing User-Defined Fields in the FDT

You can change the characteristics of user-defined fields by entering the TSS REPLACE command function shown below.

```
TSS REPLACE(FDT) FDTNAME(field-name)
 SEGMENT(segment-name) MAXLEN(nn) DISPLAY(display-name)
```

**FDTNAME** Contains an up to eight-character field-name previously defined through the TSS ADD command function.

**SEGMENT** Changes the *segment-name* associated with a *user-defined* field. You can't assign a *user-defined* field to *system-defined* segments. For a list of the predefined CA-Top Secret segments, refer to 8.2.2.1, "Defining New Fields to the FDT" on page 8-13.

- MAXLEN** Changes the number of bytes that can be used for the user-defined field, and allows one MAXLEN value per command. The total of all user-defined fields should not exceed 32,767 bytes.
- DISPLAY** Changes an up to 11-character *display-name* that is associated with a particular field and segment in the FDT Record, and allows one *display-name* per command that can be a letter, number, or special character. The *display-name* should be enclosed in single quotes if it contains blanks.

**Example:**

To change the maximum length for the HPHONE field (created on 8-14) from 12 to 14 bytes by entering the command shown below.

```
TSS REP(FDT) FDTNAME(HPHONE) MAXLEN(14)
```

### 8.2.2.3 Removing Fields From the FDT

You can remove a user-defined field by entering the TSS REMOVE command function shown below.

```
TSS REMOVE(FDT) FDTNAME(field-name)
```

- FDTNAME** Removes an up to eight-character user-defined field from the FDT Record when FDT is the target ACID name, and allows one *field-name* per command that can be a letter, number, or special character. Only the FDTNAME is required for removing a user-defined field.

For example, to remove the HPHONE field from the FDT, enter the command shown below.

```
TSS REMOVE(FDT) FDTNAME(HPHONE)
```

### 8.2.2.4 Listing the FDT

**Listing the Entire Record:** The following TSS LIST command function lists the entire FDT including both predefined and user-defined fields.

```
TSS LIST(FDT)
```

**Listing a Field:** The following TSS LIST command function lists a specific field.

```
TSS LIST(FDT) FDTNAME(field-name)
```

**FDTNAME** Lists an up to eight-character field with its associated display and segment in the FDT Record, and allows one *field-name* per command that can be a letter, number, or special character.

For example, you can list the HPHONE field that also gives you its associated display and segment by entering the command shown below.

```
TSS LIST(FDT) FDTNAME(HPHONE)
```

**Listing a Field Code:** The following TSS LIST command function lists a specific field code which must be a user-defined field.

```
TSS LIST(FDT) FDTCODE(hex-code)
```

**FDTCODE** Lists a user-defined field code in the FDT Record when FDT is the target ACID name, and allows one *hex-code* per command with a value ranging from 01 to FF.

For example, you can list hex-code 01 by entering the command shown below.

```
TSS LIST(FDT) FDTCODE(01)
```

**Listing a Segment:** The following TSS LIST command function lists a specific segment-name and its associated fields and displays.

```
TSS LIST(FDT) SEGMENT(segment-name)
```

**SEGMENT** Lists an up to eight-character *segment-name* with its associated field and display in the FDT Record, and allows one *segment-name* per command, that can be a letter, number, or special character.

For example, to list the segment HPHNATTR to which the home area code and phone number belong, enter the command shown below.

```
TSS LIST(FDT) SEGMENT(HPHNATTR)
```

**Listing a Display:** The following TSS LIST command function lists a specific display.

```
TSS LIST(FDT) DISPLAY(display-name)
```

**DISPLAY** Lists up to an 11-character *display-name* with its associated field and segment in the FDT Record, and allows one *display-name* per command that can be a letter, number, or special character.

For example, to list the HPHNUM field to which the home area code and phone number belong, enter the command shown below.

```
TSS LIST(FDT) DISPLAY(HPHNUM)
```

You can limit the authority of an ACID to only listing the FDT, by assigning MISC8(LISTRDT) authority.

## 8.3 Static Data Table (SDT) Record

The data for Record Level Protection (RLP) and Screen Level Protection (SLP) is stored in the Static Data Table (SDT). Before you can enable RLP and SLP you must first initialize the SDT using the SDTBLOCKS parameter of TSSMAINT.

**Note:** If you are increasing the size of an existing Static Data Table, you must also extend your old SECFILE into your new SECFILE using TSSXTEND.

See the *Installation Guide* for complete instructions.

The Static Data Table (SDT) record is a reserved ACID and a Security File repository for internal, non-volatile data that is used with various PERMIT administrative functions. The SDT stores these six distinct record elements:

|                               |                                                                                                                                |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| <b>CALENDAR records</b>       | Controls access to calendars                                                                                                   |
| <b>MAP records (MAPREC)</b>   | Controls access to the MAP record associated with an OTRAN or PPT resource. MAP records support Screen Level Protection (SLP). |
| <b>MASK records (MASKREC)</b> | Controls access to a MASK record that is associated with the FCT.                                                              |
| <b>RLP records (RECORD)</b>   | Provides Record Level Protection for the FCT.                                                                                  |
| <b>SELECT records</b>         | Controls access to a SELECT record that is associated with an FCT resource.                                                    |
| <b>TIME records (TIMEREC)</b> | Controls access to the TIME record that is associated with any resource.                                                       |

These unique user-defined record IDs can be added to the SDT using the TSS ADD(SDT) command function. When CA-Top Secret is initialized, the record elements that you have currently defined will be loaded into memory. They will then be used as part of the security enforcement based on the appropriate authorizations.

Any changes to the SDT will not be reflected automatically in the in-core storage copy of any modified element since several commands may be needed to build or modify an SDT record element. When completed, the record elements can be refreshed in in-core storage using a TSS MODIFY command.

**Administrative Authority:** You must have MISC3(SDT) authority to add to, remove, or list the SDT Record or the fields within it. MISC8(LISTSMT) can be used to limit the administrator's authority to only listing the SDT Record.

**Note:** A detailed procedure for implementing Record Level Protection (RLP) and Screen Level Protection (SLP) can be found in Chapter 7, Protecting Resources.

### 8.3.1 Defining Record Elements to the SDT

As stated in the introduction, there are six record elements that you can define to the SDT. They are:

- CALENDAR records
- MAP records (MAPREC)
- MASK records (MASKREC)
- RPL records (RECORD)
- SELECT records
- TIME records (TIMEREC)

Each record element is defined using the TSS ADD(SDT) command function. Since each record element has different keywords associated with it, separate sections are used to explain how to define each type of record element.

#### 8.3.1.1 How to Define CALENDAR Records

To define a new CALENDAR record to the SDT, use the command shown below.

```
TSS ADD(SDT) CALENDAR(cal-name) DESCRIPT(descript-name)
 YEAR(yyyy) DAYS(days,...)
 EXCLUDE(mm/dd,mm/dd,...) INCLUDE(mm/dd,mm/dd,...)
```

| <b>Operand</b>       | <b>Description</b>                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>cal-name</b>      | Specifies an eight-character, user-defined record ID that must be unique for each calendar. It can contain letters, numbers, and special characters.                                                                                                                                                                                                                                                                                   |
| <b>descript-name</b> | Designates an optional 32-character, user-description field that will be used as a logical name for this record.<br><br>If the description field contains blanks, you must enclose it in single quotes.                                                                                                                                                                                                                                |
| <b>yyyy</b>          | Specifies the year in which this calendar record is active. If not specified, the current year is used (the default).<br><br><b>Note:</b> A defined calendar expires at the end of the year (12/31/xx). To keep the calendar active, you must redefine it specifying the new year for yyyy. If the new calendar is not defined by January 1, then the permission will no longer be valid and the expected access will no longer exist. |
| <b>days</b>          | Indicates which days to include in the calendar. Valid entries are: SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, WEEKDAYS, and WEEKENDS.                                                                                                                                                                                                                                                                            |



When defining a calendar, it is important to remember that all days to be included in the calendar must be explicitly specified. For example, the command `TSS ADD(SDT) CALENDAR(CAL1)` will create a calendar, but there will be no valid days to use it. If you want to create a calendar that includes all days, you must specify every day as shown in the following command.

```
TSS ADD(SDT) CALENDAR(CAL1) DAYS(WEEKDAYS,WEEKENDS)
```

**mm/dd**

When used with `EXCLUDE`, lists specific dates that are to be excluded from the calendar. These dates are in addition to the days specified on the `DAYS` keyword.

When used with `INCLUDE`, lists specific dates that are to be included in the calendar. These dates are in addition to the days specified on the `DAYS` keyword.

**Examples:** To create a calendar for the present year called CAL1 that includes the days Monday through Friday, enter:

```
TSS ADD(SDT) CALENDAR(CAL1) DAYS(WEEKDAYS)
```

To create a calendar for the present year with a user-description field called PAYROLL CALENDAR, enter:

```
TSS ADD(SDT) CALENDAR(CAL1) DESCRIPT('PAYROLL CALENDAR')
DAYS(WEEKDAYS)
```

To create a 1996 calendar named FIN98, enter:

```
TSS ADD(SDT) CALENDAR(FIN8) YEAR(1998) DAYS(WEEKDAYS)
```

To create a calendar for the present year with every Monday, Wednesday, Thursday and Friday enabled, enter:

```
TSS ADD(SDT) CALENDAR(CAL1) DAYS(MON.WED,THUR,FRI)
```

To create a calendar for the present year with April 16 disabled, enter:

```
TSS ADD(SDT) CALENDAR(CAL1) DAYS(WEEKDAYS,WEEKENDS) EXCL(04/16)
```

To create a calendar for the present year with April 22 and April 29 enabled, enter:

```
TSS ADD(SDT) CALENDAR(CAL1) INCL(04/22,04/29)
```

### 8.3.1.2 How to Define MAP Records

To define a new MAP record to the SDT, use the command shown below.

```
TSS ADD(SDT) MAPREC(map-name) DESCRIPT(descript-name)
 MAPDATA(mapdata-fld1,...mapdata-fld5)
```

**Note:** The MAPREC keyword is **required** when adding a MAP record to the SDT Record.

| <b>Operand</b>           | <b>Description</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |                 |                                                                                                                                          |             |                                                                                                                                              |            |                                              |               |                                                 |               |                                                     |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------------------------------------------------------------------------------------------------------------------------------------------|------------|----------------------------------------------|---------------|-------------------------------------------------|---------------|-----------------------------------------------------|
| <b>map-name</b>          | Specifies an eight-character, user-defined record ID that must be unique for each MAP record. The name can contain letters, numbers, and special characters.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                 |                                                                                                                                          |             |                                                                                                                                              |            |                                              |               |                                                 |               |                                                     |
| <b>descript-name</b>     | Designates an optional 32-character, user-description field that will be used as a logical name for this record.<br><br>If the description field contains blanks, you must enclose it in single quotes.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                 |                                                                                                                                          |             |                                                                                                                                              |            |                                              |               |                                                 |               |                                                     |
| <b>mapdata-fld1 to 5</b> | Specifies the layout of this screen field. You can specify up to five fields on each MAPDATA keyword. Each <i>mapdata-fldn</i> operand consists of five sub-fields: <table border="0" style="margin-left: 20px;"> <tr> <td><b>fld-name</b></td> <td>Specifies an up to 24-character field name unique within this MAP record. The name can contain letters, numbers, and special characters.</td> </tr> <tr> <td><b>type</b></td> <td>Indicates the field type. Replace <i>type</i> with one of these values: CHAR (character), BIN (binary), PACKED, ZONED, or HEX (hexadecimal).</td> </tr> <tr> <td><b>row</b></td> <td>Specifies a numeric row for this data field.</td> </tr> <tr> <td><b>column</b></td> <td>Specifies a numeric column for this data field.</td> </tr> <tr> <td><b>length</b></td> <td>Specifies the length of this data field (optional).</td> </tr> </table> | <b>fld-name</b> | Specifies an up to 24-character field name unique within this MAP record. The name can contain letters, numbers, and special characters. | <b>type</b> | Indicates the field type. Replace <i>type</i> with one of these values: CHAR (character), BIN (binary), PACKED, ZONED, or HEX (hexadecimal). | <b>row</b> | Specifies a numeric row for this data field. | <b>column</b> | Specifies a numeric column for this data field. | <b>length</b> | Specifies the length of this data field (optional). |
| <b>fld-name</b>          | Specifies an up to 24-character field name unique within this MAP record. The name can contain letters, numbers, and special characters.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                 |                                                                                                                                          |             |                                                                                                                                              |            |                                              |               |                                                 |               |                                                     |
| <b>type</b>              | Indicates the field type. Replace <i>type</i> with one of these values: CHAR (character), BIN (binary), PACKED, ZONED, or HEX (hexadecimal).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                 |                                                                                                                                          |             |                                                                                                                                              |            |                                              |               |                                                 |               |                                                     |
| <b>row</b>               | Specifies a numeric row for this data field.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                 |                                                                                                                                          |             |                                                                                                                                              |            |                                              |               |                                                 |               |                                                     |
| <b>column</b>            | Specifies a numeric column for this data field.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |                 |                                                                                                                                          |             |                                                                                                                                              |            |                                              |               |                                                 |               |                                                     |
| <b>length</b>            | Specifies the length of this data field (optional).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                 |                                                                                                                                          |             |                                                                                                                                              |            |                                              |               |                                                 |               |                                                     |

**Examples:** To create a MAP record called ENG1 in the SDT, enter:

```
TSS ADD(SDT) MAPREC(ENG1)
```

To add a MAPDATA field to the MAP record ENG1 that contains a salary field in binary format called PAY which is in the 5th row and 6th column, and has a length eight characters, enter:

```
TSS ADD(SDT) MAPREC(ENG1) MAPDATA(PAY,BIN,5,6,8)
```

### 8.3.1.3 How to Define MASK Records

To define a new MASK record to the SDT, use the command shown below.

```
TSS ADD(SDT) MASKREC(mask-name) DESCRIPT(descript-name)
 MASKDATA(maskdata-fld1,...maskdata-fld5)
```

**Note:** The MASKREC keyword is **required** when adding a MASK record to the SDT Record.

| <b>Operand</b>            | <b>Description</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                 |                                                                                                                                           |             |                                                                                                                                          |               |                                             |               |                                                 |             |                                                                                                                                                                      |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------|-------------|------------------------------------------------------------------------------------------------------------------------------------------|---------------|---------------------------------------------|---------------|-------------------------------------------------|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>mask-name</b>          | Specifies an eight-character, user-defined record ID that must be unique for each MASK record. The name can contain letters, numbers, and special characters.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |                 |                                                                                                                                           |             |                                                                                                                                          |               |                                             |               |                                                 |             |                                                                                                                                                                      |
| <b>descript-name</b>      | Designates an optional 32-character, user-description field that will be used as a logical name for this record.<br><br>If the description field contains blanks, you must enclose it in single quotes.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                 |                                                                                                                                           |             |                                                                                                                                          |               |                                             |               |                                                 |             |                                                                                                                                                                      |
| <b>maskdata-fld1 to 5</b> | Contains the layout of the field to be masked. You can specify up to five fields on each MASKDATA keyword. Each <i>maskdata-fldn</i> operand consists of five sub-fields: <table border="0" style="margin-left: 20px;"> <tr> <td><b>fld-name</b></td> <td>Specifies an up to 24-character field name unique within this MASK record. The name can contain letters, numbers, and special characters.</td> </tr> <tr> <td><b>type</b></td> <td>Indicates the field type. Replace <i>type</i> with one of these values: CHAR (character), BIN (binary), PACKED, ZONED, HEX(hexadecimal).</td> </tr> <tr> <td><b>offset</b></td> <td>Indicates an up to five-digit offset value.</td> </tr> <tr> <td><b>length</b></td> <td>Indicates an up to 44-decimal value for length.</td> </tr> <tr> <td><b>mask</b></td> <td>Indicates an up to 44-character mask value. This value can contain letters, numbers, and special characters; however, the value must match the field type. The value</td> </tr> </table> | <b>fld-name</b> | Specifies an up to 24-character field name unique within this MASK record. The name can contain letters, numbers, and special characters. | <b>type</b> | Indicates the field type. Replace <i>type</i> with one of these values: CHAR (character), BIN (binary), PACKED, ZONED, HEX(hexadecimal). | <b>offset</b> | Indicates an up to five-digit offset value. | <b>length</b> | Indicates an up to 44-decimal value for length. | <b>mask</b> | Indicates an up to 44-character mask value. This value can contain letters, numbers, and special characters; however, the value must match the field type. The value |
| <b>fld-name</b>           | Specifies an up to 24-character field name unique within this MASK record. The name can contain letters, numbers, and special characters.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |                 |                                                                                                                                           |             |                                                                                                                                          |               |                                             |               |                                                 |             |                                                                                                                                                                      |
| <b>type</b>               | Indicates the field type. Replace <i>type</i> with one of these values: CHAR (character), BIN (binary), PACKED, ZONED, HEX(hexadecimal).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                 |                                                                                                                                           |             |                                                                                                                                          |               |                                             |               |                                                 |             |                                                                                                                                                                      |
| <b>offset</b>             | Indicates an up to five-digit offset value.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                 |                                                                                                                                           |             |                                                                                                                                          |               |                                             |               |                                                 |             |                                                                                                                                                                      |
| <b>length</b>             | Indicates an up to 44-decimal value for length.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                 |                                                                                                                                           |             |                                                                                                                                          |               |                                             |               |                                                 |             |                                                                                                                                                                      |
| <b>mask</b>               | Indicates an up to 44-character mask value. This value can contain letters, numbers, and special characters; however, the value must match the field type. The value                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |                 |                                                                                                                                           |             |                                                                                                                                          |               |                                             |               |                                                 |             |                                                                                                                                                                      |

specified for *mask* will appear in place of the actual value.

**Example:** To add a MASKDATA field to the MASK record CRYPT1 that contains a salary field in character format called ADDR, masked with an asterisk (\*), having an offset value of 50 and a length of 20 characters, enter:

```
TSS ADD(SDT) MASKREC(CRYPT1) MASKDATA(ADDR,CHAR,50,20,**)
```

### 8.3.1.4 How to Define RLP Records

To define a new RLP record to the SDT, use the command shown below.

```
TSS ADD(SDT) RECORD(rlp-name) DESCRIPT(descript-name)
 RECDATA(recdata-fld1,...recdata-fld5)
```

**Note:** The RECORD keyword is **required** when adding a RLP record to the SDT Record.

| <b>Operand</b>           | <b>Description</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                 |                                                                                                                                      |             |                                                                                                                                             |               |                                                         |               |                                                                  |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|--------------------------------------------------------------------------------------------------------------------------------------|-------------|---------------------------------------------------------------------------------------------------------------------------------------------|---------------|---------------------------------------------------------|---------------|------------------------------------------------------------------|
| <b>rlp-name</b>          | Specifies an eight-character, user-defined record ID that must be unique for each RLP record. It can contain letters, numbers, and special characters. In addition, it must match the name of the FCT being protected by the RLP feature.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                 |                                                                                                                                      |             |                                                                                                                                             |               |                                                         |               |                                                                  |
| <b>descript-name</b>     | Designates an optional 32-character, user-description field that will be used as a logical name for this record.<br><br>If the description field contains blanks, you must enclose it in single quotes.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                 |                                                                                                                                      |             |                                                                                                                                             |               |                                                         |               |                                                                  |
| <b>recdata-fld1 to 5</b> | Contains the layout a field in this record. You can specify up to five fields on each RECDATA keyword. Each <i>recdata-fldn</i> operand consists of five sub-fields: <table border="0" style="margin-left: 20px;"> <tr> <td style="padding-left: 10px;"><b>fld-name</b></td> <td>Specifies an up to 24-character field name unique to this RLP record. The name can contain letters, numbers, and special characters.</td> </tr> <tr> <td style="padding-left: 10px;"><b>type</b></td> <td>Indicates the field type. Replace <i>type</i> with one of these values: CHAR (character), BIN (binary), PACKED, ZONED or HEX (hexadecimal).</td> </tr> <tr> <td style="padding-left: 10px;"><b>offset</b></td> <td>Specifies an up to five-digit offset of the data field.</td> </tr> <tr> <td style="padding-left: 10px;"><b>length</b></td> <td>Specifies an up to 44-digit length of the data field (optional).</td> </tr> </table> | <b>fld-name</b> | Specifies an up to 24-character field name unique to this RLP record. The name can contain letters, numbers, and special characters. | <b>type</b> | Indicates the field type. Replace <i>type</i> with one of these values: CHAR (character), BIN (binary), PACKED, ZONED or HEX (hexadecimal). | <b>offset</b> | Specifies an up to five-digit offset of the data field. | <b>length</b> | Specifies an up to 44-digit length of the data field (optional). |
| <b>fld-name</b>          | Specifies an up to 24-character field name unique to this RLP record. The name can contain letters, numbers, and special characters.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                 |                                                                                                                                      |             |                                                                                                                                             |               |                                                         |               |                                                                  |
| <b>type</b>              | Indicates the field type. Replace <i>type</i> with one of these values: CHAR (character), BIN (binary), PACKED, ZONED or HEX (hexadecimal).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |                 |                                                                                                                                      |             |                                                                                                                                             |               |                                                         |               |                                                                  |
| <b>offset</b>            | Specifies an up to five-digit offset of the data field.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                 |                                                                                                                                      |             |                                                                                                                                             |               |                                                         |               |                                                                  |
| <b>length</b>            | Specifies an up to 44-digit length of the data field (optional).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                 |                                                                                                                                      |             |                                                                                                                                             |               |                                                         |               |                                                                  |

**Examples:** To create an RLP record called PROLL1, and a RECDATA field that contains salary information in character format with a length of six characters and an offset of 48. enter:

```
TSS ADD(SDT) RECORD(PROLL1) RECDATA(PAY,CHAR,48,6)
```

### 8.3.1.5 How to Define SELECT Records

To define a new SELECT record to the SDT, use the command shown below.

```
TSS ADD(SDT) SELECT(sel-name) DESCRIPT(desc-name)
 SELDATA('IF seldata-flid {logical operator} "value"
 [AND|OR] seldata-flid {logical operator} "value"')
```

**Note:** The SELECT keyword is **required** when adding a SELECT record to the SDT Record.

| <b>Operand</b>          | <b>Description</b>                                                                                                                                                                                      |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>sel-name</b>         | Specifies an eight-character, user-defined record ID that must be unique for each SELECT record. It can contain letters, numbers, and special characters.                                               |
| <b>descript-name</b>    | Designates an optional 32-character, user-description field that will be used as a logical name for this record.<br><br>If the description field contains blanks, you must enclose it in single quotes. |
| <b>seldata-flid</b>     | Contains the logical rules to be used as selection criteria for this expression. <i>seldata-flid</i> must be surrounded by single quotes.                                                               |
| <b>logical operator</b> | Select one of these logical operators: EQ (equal to), NE (not uqual to), LT (less than), GT (greater than), GE (greater than or equal to), LE (less than or equal to).                                  |
| <b>value</b>            | Specify a numerical value to which the value of <i>seldata-flid</i> is to be compared. <i>value</i> must be surrounded by double quotes.                                                                |
| <b>AND OR</b>           | Used to link multiple statements together. Use AND when <b>both</b> statements are to be true; use OR when <b>either</b> statement is to be true.                                                       |

**Examples:** To create a SELECT record called PROBE1 in the SDT, selecting departments ranging from 200 through 299, enter:

```
TSS ADD(SDT) SELECT(PROBE1)
 SELDATA('IF DEPT GE "200" AND DEPT LE "299" ')
```

### 8.3.1.6 How to Define TIME Records

To define a new TIME record to the SDT, use the command shown below.

```
TSS ADD(SDT) TIMEREC(time-name) DESCRIPT(descript-name)
 RANGE(hmm,hmm,...)
```

**Note:** The TIMEREC keyword is **required** when adding a TIME record to the SDT Record.

| <b>Operand</b>   | <b>Description</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>time-name</b> | Specifies an eight-character <i>time-name</i> that is user-defined and can contain letters, numbers or special characters.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>hhmm:hhmm</b> | Specifies one, or a series of, entries in the format hhmm:hhmm for the starting and ending times of a range. You can indicate up to 53 ranges per command.<br><br>The value for minutes must be 00, 15, 30, or 45— designating a 15 minute block of time. For example, 00 refers to the minutes 00 to 14; therefore, RANGE(1200:1300) signifies 12 noon through 1:14 p.m. If you only want noon to 1 p.m., you must specify RANGE(1200:1245) to designate those four quarter hours.<br><br>In addition, the end time must be greater than the start time, and wrapping is not allowed. For example, if you want to specify 11 p.m. until 1 a.m., enter:<br><br>RANGE(2300:2345,0000:0045) <b>not</b> RANGE(2300:0045)<br><br>which would be invalid. |

**Examples:** To add a TIME record called TEMP1 to the SDT, enter:

```
TSS ADD(SDT) TIMEREC(TEMP1)
```

To add a time period from 1 p.m. to 5 p.m. to the RANGE field of the TIME record called TEMP1 in the SDT, enter:

```
TSS ADD(SDT) TIMEREC(TEMP1) RANGE(1300:1645)
```

## 8.3.2 Replacing Data Elements in the SDT

You can replace any of the record elements that you have defined to the SDT by using the TSS REPLACE(SDT) command function. The same keywords that were used to define each record element on the TSS ADD(SDT) command are used to replace the values. The next sections explain how to replace the values for each record element.

### 8.3.2.1 How to Replace CALENDAR Records

To replace an existing CALENDAR record with a new dates, enter:

```
TSS REPLACE(SDT) CALENDAR(cal-name)
 YEAR(yyyy) DAYS(days,...)
 EXCLUDE(mm/dd,mm/dd,...) INCLUDE(mm/dd,mm/dd,...)
```

### 8.3.2.2 How to Replace MAP Records

To replace an existing MAP record with a new field, enter:

```
TSS REPLACE(SDT) MAPREC(map-name) MAPDATA(fld-name)
```

### 8.3.2.3 How to Replace MASK Records

To replace an existing MASK record with a new field, enter:

```
TSS REPLACE(SDT) MASKREC(mask-name)
 MASKDATA(maskdata-fld1,...maskdata-fld2)
```

### 8.3.2.4 How to Replace RLP Records

To replace an existing RLP record with a new field, enter:

```
TSS REPLACE(SDT) RECORD(rlp-name) RECDATA(fld-name)
```



### 8.3.2.5 How to Replace SELECT Records

To replace an existing SELECT record with a new field, enter:

```
TSS REPLACE SELECT(sel-name)
 SELDATA('IF seldata-flid {logical operator} "value"
 [AND|OR] seldata-flid {logical operator} "value"')
```

### 8.3.2.6 How to Replace TIME Records

To replace an existing TIME record with a new range, enter:

```
TSS REPLACE(SDT) TIMEREC(time-name) RANGE(hhmm,hhmm,...)
```

## 8.3.3 Removing Fields From Records in the SDT

You can remove fields in any of the record elements that you have defined to the SDT by using the TSS REMOVE(SDT) command function. The next sections explain how to remove the values for each record element.

### 8.3.3.1 How to Remove Dates From CALENDAR Records

To remove only specified dates from an existing CALENDAR record, enter:

```
TSS REMOVE(SDT) CALENDAR(cal-name)
 YEAR(yyyy) DAYS(days,...)
 EXCLUDE(mm/dd,mm/dd,...) INCLUDE(mm/dd,mm/dd,...)
```

### 8.3.3.2 How to Remove Fields From MAP Records

To remove up to five fields from an existing MAP record, enter:

```
TSS REMOVE(SDT) MAPREC(map-name)
 SDTFNAME(mapdata-flid1,...mapdata-flid5)
```

### 8.3.3.3 How to Remove Fields From MASK Records

To remove up to five fields from an existing MASK record, enter:

```
TSS REMOVE(SDT) MASKREC(mask-name)
 SDTFNAME(maskdata-fld1,...maskdata-fld5)
```

### 8.3.3.4 How to Remove Fields From RLP Records

To remove up to five fields from an existing RLP record, enter:

```
TSS REMOVE(SDT) RECORD(rlp-name)
 SDTFNAME(recdata-fld1,...recdata-fld5)
```

### 8.3.3.5 How to Remove Fields From SELECT Records

To remove up to five fields from an existing SELECT record, enter:

```
TSS REMOVE(SDT) SELECT(sel-name) SDTFNAME(seldata-fld1,...seldata-fld5)
```

### 8.3.3.6 How to Remove Times From TIME Records

To remove only specified times from an existing TIME record, enter:

```
TSS REMOVE(SDT) TIMEREC(time-name)
```

## 8.3.4 Deleting Record Elements From the SDT

You can delete any record element that you have defined to the SDT by using the TSS DELETE(SDT) command function. The next sections explain how to delete each record element.

#### 8.3.4.1 How to Delete CALENDAR Records

To delete a specified CALENDAR record from the SDT, enter:

```
TSS DELETE(SDT) CALENDAR(cal-name)
```

#### 8.3.4.2 How to Delete MAP Records

To delete a specified MAP record from the SDT, enter:

```
TSS DELETE(SDT) MAPREC(map-name)
```

#### 8.3.4.3 How to Delete MASK Records

To delete a specified MASK record from the SDT, enter:

```
TSS DELETE(SDT) MASKREC(mask-name)
```

#### 8.3.4.4 How to Delete RLP Records

To delete a specified RLP record from the SDT, enter:

```
TSS DELETE(SDT) RECORD(rlp-name)
```

#### 8.3.4.5 How to Delete SELECT Records

To delete a specified SELECT record from the SDT, enter:

```
TSS DELETE(SDT) SELECT(sel-name)
```

### 8.3.4.6 How to Delete TIME Records

To delete a specified TIME record from the SDT, enter:

```
TSS DELETE(SDT) TIMEREC(time-name)
```

## 8.3.5 Listing Record Elements in the STD

You can list any record element that you have defined to the SDT by using the TSS LIST(SDT) command function. The next sections explain how to list each record element.

You can limit a user's authority to only listing the records in the STD by granting them MISC8(SDT) authority.

### 8.3.5.1 How to List CALENDAR Records

To list all or only a specified CALENDAR record from the SDT, enter:

```
TSS LIST(SDT) CALENDAR(ALL|cal-name)
```

### 8.3.5.2 How to List MAP Records

To list all or only a specified MAP record from the SDT, enter:

```
TSS LIST(SDT) MAPREC(ALL|map-name)
```

### 8.3.5.3 How to List MASK Records

To list all or only a specified MASK record from the SDT, enter:

```
TSS LIST(SDT) MASKREC(ALL|mask-name)
```

#### 8.3.5.4 How to List RLP Records

To list all or only a specified RLP from the SDT, enter:

```
TSS LIST(SDT) RECORD(ALL|r1p-name)
```

#### 8.3.5.5 How to List SELECT Records

To list all or only a specified SELECT from the SDT, enter:

```
TSS LIST(SDT) SELECT(ALL|sel-name)
```

#### 8.3.5.6 How to List TIME Records

To list all or only a specified TIME record from the SDT, enter:

```
TSS LIST(SDT) TIMEREC(ALL|time-name)
```

## 8.4 Node Descriptor Table (NDT) Record

The Node Descriptor (NDT) is a reserved or special ACID and global record similar to the Resource Descriptor Table and the Field Descriptor Table that contains data for assigning PassTickets and Session Keys to applications. It also contains VAX-related data such as—UAF, Node, NETUAF, and Volume records. Refer to the *CA-TopSecret/Secman for VAX Implementation Guide* for more VAX-related information.

**Administrative Authority:** You must have MISC1(NDT) authority to maintain data in the NDT.

### 8.4.1 Defining Data to the NDT

**Defining Session Keys to Applications:** To assign a session key to an application use the command shown below.

```
TSS ADD(NDT) PSTKAPPL(application) SESSKEY(session-key)
```

The following two keywords are required when adding session keys to applications.

**PSTKAPPL** Identifies an up to eight-character *application* that is assigned a session key for PassTicket processing, and allows one *application* per command that can be a letter, number, or special character.

**SESSKEY** Specifies an up to 16-character hexadecimal "password" that is unique to each application defined by a PSTKAPPL keyword. You **must** supply a SESSKEY with PSTKAPPL.

**Example:** To indicate that the session key for KA180987 is A1B2C3, enter the command shown below.

```
TSS ADD(NDT) PSTKAPPL(KA180987) SESSKEY(A1B2C3)
```

**Defining VAX-Related Data:** To assign VAX-related data such as a VXNODE to the NDT, enter the command shown below.

```
TSS ADD(NDT) VXNODE(vaxnode-name)
```

**VXNODE** Identifies an up to six-character VAX prefix *node-name* to the NDT, and allows one *node-name* per command.

For example, to assign the VAX node USDC01 to the NDT, enter the command shown below.

```
TSS ADD(NDT) VXNODE(USDC01)
```

## 8.4.2 Changing Values in the NDT

**Changing Values of an Application:** To replace an application, enter the command shown below.

```
TSS REP(NDT) PSTKAPPL(application)
```

**PSTKAPPL** Changes an up to eight-character *application* that maintains the same session key for PassTicket processing, and allows one *application* per command that can be a letter, number, or special character.

For example, to replace application KA180987 with application KA180999, enter the command shown below.

```
TSS REP(NDT) PSTKAPPL(KA180999)
```

**Changing Values of VAX-Related Data:** To replace VAX-related data such as a VXNODE, enter the command shown below.

```
TSS REP(NDT) VXNODE(node-name)
```

**VXNODE** Changes an up to six-character VAX prefix *node-name* to the NDT, and allows one *node-name* per command that can be a number, letter, or special character.

For example, to change the VAX node USDC01 to USDC02, enter the command shown below.

```
TSS REP(NDT) VXNODE(USDC02)
```



### 8.4.3 Removing Data From the NDT

**Removing an Application:** To remove an application and its accompanying session key, use the command shown below.

```
TSS REM(NDT) PSTKAPPL(application)
```

The following keyword is required when removing an application.

**PSTKAPPL** Removes an up to eight-character *application* and its associated session key used for PassTicket processing, and allows one *application* per command that can be a letter, number, or special character.

For example, to remove the application KA180987 and its associated session key, enter the following command.

```
TSS REM(NDT) PSTKAPPL(KA180987)
```

**Removing VAX-Related Data:** To remove VAX-related data such as a VXNODE from the NDT, enter the command shown below.

```
TSS REM(NDT) VXNODE(vaxnode-name)
```

**VXNODE** Removes an up to six-character VAX prefix *node-name* from the NDT, and allows one *node-name* per command.

For example, to remove the VAX node USDC01 from the NDT, enter the command shown below.

```
TSS REM(NDT) VXNODE(USDC01)
```

### 8.4.4 Listing the NDT

**Listing the Entire Record:** You can list the entire NDT Record by issuing the command shown below.

```
TSS LIST(NDT) DATA(ALL)
```

**Listing Session Keys and Applications:** To list session keys and associated applications, enter the command shown below.

```
TSS LIST(NDT) DATA(SESSKEY)
```

**DATA** Specifies the particular data type that is listed. For session keys, you must enter SESSKEY.

**Listing VAX-Related Data:** To list VAX-related data such as VXNODE, enter the command shown below.

```
TSS LIST(NDT) DATA(VXNODE)
```

**DATA** Specifies the particular data type that is listed. For a VAX node, you must enter VXNODE.

To list VAX-related data such as VXNODE with a particular node-name, enter the command shown next.

```
TSS LIST(NDT) DATA(VXNODE) ITEM(node-name)
```

**DATA** Specifies the particular data type that is listed. For a VAX node, you must enter VXNODE.

**ITEM** Specifies a particular node-name, uaf-name, device-name, or netuaf-name.

For example, to list VXNODE USDC02, enter the command shown below.

```
TSS LIST(NDT) DATA(VXNODE) ITEM(USDC02)
```

## 8.5 ALL Record

The ALL Record is a special or reserved ACID that identifies resources globally accessible to all users. The following command function, which uses ALL as both an ACID and access level, gives ALL users ALL access to the resource entered.

```
TSS PERMIT(ALL) resource(operand) ACCESS(ALL)
```

**Note:** The ALL reserved ACID is only used with the TSS ADD or REMOVE command function for the OPERCMDS, COMMAND, FACILITY, XCOMMAND, and USER keywords.

**Administrative Authority:** You must have MISC9(GLOBAL) authority to assign administrative authorities to the ALL Record.

**Permitting a Data Set:** The data set UNSOLD.MASTER.FILE is permitted to the ALL Record with ALL access by entering the command shown below.

```
TSS PERMIT(ALL) DSN('UNSOLD.MASTER.FILE') ACCESS(ALL)
```

**Permitting a Facility:** To permit all users to access the TSO facility between the hours of 9:00 a.m. and 5:59 p.m., enter the command shown below.

```
TSS ADD(ALL) FAC(TSO) TIME (09,17)
```

**Permitting Operator Commands:** To permit MVS commands issued from a system console to the ALL Record, enter the command shown below.

```
TSS ADD(ALL) OPERCMDS(MVS.)
```

## 8.6 APPCLU Record

The APPCLU Record is a reserved or special ACID that identifies which logical units (LUs) can establish a link for processing APPC transactions and conversations.

For more details on the APPCLU Record, refer to the CA-Top Secret *Implementation: BATCH, STC and APPC Guide*.

**Administrative Authority:** You must have MISC2(APPCLU) authority to add specific LINKIDs within the APPCLU Record.

### Authority

### 8.6.1 Defining LINKIDs to the APPCLU Record

The LINKID keyword, shown below, is used to identify which LUs can be used for APPC conversation processing shown below.

```

TSS ADD(APPCLU) LINKID(netid.locallu.remotelu)

 [SESSKEY(nnnnnnnn)]
 [INTERVAL(nnnnn)]
 []
 [CONVSEC { (NONE) }]
 [{ (ALREADYV) }]
 [{ (CONV) }]
 [{ (PERSISTV) }]
 [{ (AVPV) }]
 []
 [SESSLOCK]

```

**netid** Identifies the network on which the local LU resides. This value should be derived from the value specified in the "netid=" statement of the VTAM ATCSTR member.

**locallu** Identifies the name of the local LU.

**remotelu** Identifies the partner LU.

Prefixing is supported; masking is not. For more details on prefixing and masking refer to Chapter 6, *Resource Security Validation*.

**Note:** The Network Qualified Names (NQN) option, introduced with VTAM 4.1, requires the LINKID to include a remote netid qualifier. The format of such a LINKID is shown below:

```
localnetid.locallu.remotenetid.remotelu
```

This format must be used if NQN is on. In contrast, if NQN is off, the original three-qualifier format should be used. For more information concerning the CA-Top Secret algorithm for selecting the "best match", refer to the *Implementation: Batch and APPC Guide*.

The following keywords can be used with LINKID:

|                 |                                                                                                                                                                           |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>SESSKEY</b>  | Identifies an up to 16-byte hexadecimal "password" used to verify the link when security is in effect.                                                                    |
| <b>INTERVAL</b> | Identifies the number of days for which the SESSKEY is valid. The value for INTERVAL can range from 0 to 32767. Zero indicates that the SESSKEY won't need to be changed. |
| <b>CONVSEC</b>  | Determines what security information needs to be validated when a conversation request is received. The following operands are used with the CONVSEC keyword:             |
| <b>NONE</b>     | Indicates that no security validation will be performed.                                                                                                                  |
| <b>ALREADYV</b> | Indicates that security validation has already occurred. A valid userid and password, however, are still required.                                                        |
| <b>CONV</b>     | Indicates security information is required. A userid and password need to be verified.                                                                                    |
| <b>PERSISTV</b> | Indicates that a userid and password must be verified on the first request; on subsequent requests only the userid is verified.                                           |
| <b>AVPV</b>     | Supports both ALREADYV and PERSISTV values the security type used depends on the incoming request.                                                                        |
| <b>SESSLOCK</b> | Indicates that these particular LUs aren't authorized to be used for APPC conversations.                                                                                  |

**Example:** You want to establish a link between LU01 and LU02. When a TP on LU01 initiates a conversation request with a TP on LU02, the SESSKEY and the initiating ACID must be validated. The SESSKEY must be changed every 30 days. Issue the command shown below to fulfill these requirements.

```
TSS ADD(APPCLU) LINKID(SYS1.LU01.LU02) CONVSEC(CONV)
SESSKEY(1234) INTERVAL(30)
```

To establish a similar task with NQN on, the command should read as follows:

```
TSS ADD(APPCLU) LINKID(SYS1.LU01.SYS2.LU02) CONVSEC(CONV)
SESSKEY(1234) INTERVAL(30)
```

## 8.6.2 Changing Data in the APPCLU Record

You can change or remove data in the APPCLU Record by using the command shown below.

```
TSS REPLACE(APPCLU) LINKID(netid.locallu.remotelu)
[SESSKEY(nnnnnnnn)]
[INTERVAL(nnnnn)]
[]
[CONVSEC { (NONE) }]
[{ (ALREADYV) }]
[{ (CONV) }]
[{ (PERSISTV) }]
[{ (AVPV) }]
[SESSLOCK]
```

Refer to 8.6.1, “Defining LINKIDs to the APPCLU Record” on page 8-40, for a description of all the keywords associated with LINKID in the APPCLU Record.

**Changing Data:** You can use all of the keywords associated with LINKID to change data in the APPCLU Record.

For example, you want to change the SESSKEY interval to every 15 days on the link that you established previously between LU01 and LU02 by issuing the command shown below.

```
TSS REP(APPCLU) LINKID(SYS1.LU01.LU02)
SESSKEY(1234) INTERVAL(15)
```

**Removing Data:** To remove a field from the APPCLU Record, specify the keyword followed by parentheses.

For example, you have decided to remove the conversation request keyword from the link between LU01 and LU02 by entering the command shown below.

```
TSS REP(APPCLU) LINKID(SYS1.LU01.LU02) CONVSEC()
```

### 8.6.3 Removing a Link From the APPCLU Record

A link can be removed by entering the command shown below.

```
TSS REMOVE(APPCLU) LINKID(netid.localu.remotelu)
```

For example, you want to remove a previous link between LU01 and LU02 by entering the command shown below.

```
TSS REMOVE(APPCLU) LINKID(SYS1.LU01.LU02)
```

### 8.6.4 Listing the APPCLU Record

To list the entire APPCLU Record, enter the command shown below.

```
TSS LIST(APPCLU)
```

## 8.7 AUDIT Record

The AUDIT Record is a special or reserved ACID that is used to add a specific resource for auditing purposes.

For more details on the AUDIT Record, refer to the *CA-Top Secret Auditor's Guide*.

**Administrative Authority:** When using AUDIT as a reserved ACID name, you must have RESOURCE(AUDIT) authority.

### 8.7.1 Adding a Resource to the AUDIT Record

To add a resource to the AUDIT Record, enter the command shown below.

```
TSS ADD(AUDIT) resource(operand)
```

For example, you can audit the data set UNSOLD.MASTER.FILE by entering the command shown next.

```
TSS ADD(AUDIT) DSN('UNSOLD.MASTER.FILE')
```

### 8.7.2 Removing a Resource From the AUDIT Record

To remove a resource from the AUDIT Record, enter the command shown below.

```
TSS REM(AUDIT) resource(operand)
```

For example, you can remove the data set UNSOLD.MASSTER.FILE from the AUDIT Record by entering the command below.

```
TSS REM(AUDIT) DSN('UNSOLD.MASTER.FILE')
```



### 8.7.3 Listing the AUDIT Record

**Listing the Entire Record:** To list the entire AUDIT Record, enter the command shown below.

```
TSS LIST(AUDIT)
```

**Listing a Specific Resource:** To list a specific resource in the AUDIT Record, enter the command shown below.

```
TSS LIST(AUDIT) resource(operand)
```

For example, to list the data set UNSOLD.MASTER.FILE in the AUDIT Record, enter the command shown below.

```
TSS LIST(AUDIT) DSN('UNSOLD.MASTER.FILE')
```

## 8.8 Data Lookaside Facility (DLF) Record

The Data Lookaside Facility (DLF) Record is a special or reserved ACID that controls the loading of selected data sets into ESA hiperspace by selected jobs.

For more details, refer to the CA-Top Secret *Command Functions Guide*.

**Administrative Authority:** You must have MISC2(DLF) authority to administer the DLF Record.

### 8.8.1 Defining Data to the DLF Record

To allow a specific data set to be brought into hiperspace that is accessed by one of the jobs in the JOBNAME list, enter the TSS ADD(DLF) command function.

```
TSS ADD(DLF) DSN(data-set-name) {JOBNAME(job-name1,job-name2,...)}
 {RETAIN}
```

|                |                                                                                                       |
|----------------|-------------------------------------------------------------------------------------------------------|
| <b>DSN</b>     | Identifies a two- to 26-character prefix for a <i>data-set-name</i> brought into hiperspace.          |
| <b>JOBNAME</b> | Identifies an up to eight-character <i>job-name</i> that accesses a data set brought into hiperspace. |
| <b>RETAIN</b>  | Leaves a data set in hiperspace when the job that brought it into hiperspace ends.                    |

For example, you want to add data set CICS.MASTER.FILE to the DLF Record and allow it to be brought into hiperspace when it is accessed by JOB1, and to remain in hiperspace after the job ends. To accomplish this, enter the command shown below.

```
TSS ADD(DLF) DSN('CICS.MASTER.FILE') JOBNAME(JOB1) RETAIN
```

## 8.8.2 Removing Data From the DLF Record

To remove a data set from the DLF Record, enter the command shown below.

```
TSS REM(DLF) DSN(data-set-name)
```

**DSN** Identifies a two- to 26-character prefix for a *data-set-name* brought into hyperspace.

For example, you want to remove data set CICS.MASTER.FILE from the DLF by entering the command shown below.

```
TSS REM(DLF) DSN('CICS.MASTER.FILE')
```

## 8.8.3 Listing the DLF Record

To list the entire DLF Record, enter the command shown below.

```
TSS LIST(DLF)
```



## Chapter 9. Command Propagation Facility

---

The Command Propagation Facility (CPF) lets sites administer multiple Security Files across VTAM-networked systems by propagating TSS commands, as well as user-initiated changes (such as updated passwords, permissions, maintenance, and suspensions) to all or selected nodes within that network. This process is referred to as *distributed security processing* because it allows the security administrator to distribute any change in a user's authorization rights and restrictions to any node to which that user is defined.

This chapter is designed to explain how to implement CPF for your site by providing:

- An overview of the components and CPF architecture.
- Detailed descriptions of the CPF control options and TSS command keywords.
- An in-depth discussion of the administration of default routing nodes.
- Examples of how the control options and keywords interact when a command is issued.
- A description of the CPF Recovery and Journal Files.
- A discussion of how user accountability is maintained when CPF processing is in place.

Information on the control options and command keywords can also be found in your *Control Options* and *TSS Command Functions* guides.

## 9.1 Overview

The Command Propagation Facility provides the security environment with:

- Routing of security administration to all or selected nodes within the VSE or MVS security network.
- Optional synchronous or asynchronous remote command execution.
- TSS command execution with most CA-Top Secret commands—except MODIFY, LOCK, UNLOCK, HELP, and WHOAMI.
- Automatic update of passwords on all connected systems if changed by the user during logon.
- Propagation of user-initiated suspensions for exceeding password and violation limits.
- Optional Journal Files (SYSLST, tape, or disk) to log commands transmitted to, and responses received from, remote nodes.
- Collecting asynchronous commands in an optional Recovery File so that they can be retransmitted in case of network outage.

### 9.1.1 Communication Components

To perform distributed security processing, CA-Top Secret relies on two communication components provided by the Integration Services of CA-CIS. These components are CAIENF (Event Notification Facility) and CAICCI (Common Communication Interface).

- CAIENF is an operating system interface component that offers a simple yet flexible approach for CA-Top Secret to obtain data from VSE.
- CAICCI is a common communications facility that enables CA-Top Secret secured nodes to communicate with one another. It provides the VTAM facilities needed to transmit and receive TSS commands.

**Note:** The word *node*, when used in this chapter, refers to the *unique identifier* (SYSID) that is assigned to that node when it is defined using CAICCI. A node isn't the same as the VTAM APPLID, although you can use the same names. Refer to the CA-CIS documentation for more information about defining nodes.

For more information about communication components, refer to the CA-CIS documentation set.

## 9.1.2 CPF Architecture

To use CPF, the security administrator must first be aware of all remote CPUs and the ACIDs defined to them. Most Security Files are essentially identical; those defined to one site can be defined to a remote site as well. Security Files that aren't identical can have additional ACIDs, permissions, and/or resource ownership definitions which must be taken into account. This can become a difficult task for the security administrator responsible for maintaining user information. If this is the case, you may want to consider using automatic propagation based on default nodes (DEFNODES) for ACIDs.

### 9.1.2.1 Implicit and Explicit Targeting

CPF allows you to automatically synchronize Security Files on multiple nodes through the propagation of TSS commands as well as user-initiated changes—such as suspension and password changes. Security administration propagation can be *implicit* (by using the CPF control options to set system-wide propagation rules) or *explicit* (by using the CPF command keywords to set propagation rules on a command-by-command basis).

The designated CPF control option values specified in each Parameter File determine the implicit target nodes to be used when a command is issued from that particular node. For example, if the CPF control options for NODEA identify implicit target nodes of NODEB, NODEC, and NODED, whenever a command is issued from NODEA that command is automatically sent to NODEB, NODEC, and NODED.

By using the CPF command keywords, a security administrator can override the targets designated by the control options. For example, even though the control options for NODEA identify implicit targets of NODEB, NODEC, and NODED, the security administrator can use the TARGET keyword to indicate that a particular command should only be propagated to NODEB. For more details on the TARGET keyword, refer to 9.3, “CA-Top Secret Command Keywords Used With CPF” on page 9-7.

### 9.1.2.2 Synchronous and Asynchronous Processing

In addition to designating target nodes for a command, you can also indicate how that command will be processed. By using the appropriate control options and command keywords, you can process on either a *synchronous* or *asynchronous* basis as described below.

- |                     |                                                                                                                                                                                                                                                                                     |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>synchronous</b>  | Allows the TSS command to execute simultaneously throughout the network. CA-Top Secret waits for the command response to return from the remote node before continuing.                                                                                                             |
| <b>asynchronous</b> | Allows the TSS command to execute on a selective basis throughout the network. This means that CA-Top Secret doesn't have to wait for a response from each node to resume processing. All commands transmitted through asynchronous processing are retained in a CPF Recovery File. |

These options are independent and can be used separately or together. The synchronous or asynchronous processing of commands and the specific targeted nodes are initially

determined by control option settings. Later, these values can be changed using the appropriate TSS command keywords. For more details, refer to 9.2, “CPF-Related Control Options” on page 9-5 and 9.3, “CA-Top Secret Command Keywords Used With CPF” on page 9-7.

### 9.1.2.3 Administrative Authority

In all cases, administrative authority and scope of the security administrator is verified at the sending and target nodes before the command is successfully applied to the targeted Security Files. This is an important level of control over remote administration. To enter TSS commands with a targeted destination, you need MISC2(TARGET) authority.

In addition to propagating changes, CPF allows the security administrator to view the contents of Security Files in remote nodes. This viewing is completely secure since scope is verified at both locations, allowing the administrator or auditor to review the security information for which he or she is responsible at all nodes in the CPF domain. CPF also encrypts the data that it sends between nodes.

Propagated administration commands execute on the remote system using the authority and scope of the user as defined on the remote system, and not from the originating system. For example, if a user who is defined as an SCA on one system propagates a TSS command to a system where that user is defined only as a USER, then the command will be limited to the USER authority.

User initiated (versus administrator initiated) password changes propagated through CPF cause the user's password to change at each node where the change is sent, provided that the user's existing passwords are the same. The password won't change at nodes where the existing passwords aren't identical or aren't synchronized. This prevents one user from changing the password of an identically named ACID on another node that is used by another person.



## 9.2 CPF-Related Control Options

CA-Top Secret supplies control options that govern the use of CPF and enable distributed security to be maintained efficiently. At least one of the CPF-related control options described below **must** be entered at CA-Top Secret startup to use the Command Propagation Facility. If it isn't, CPF can't be activated until the next CA-Top Secret startup, and no CPF control options will be accepted by CA-Top Secret until that time. Once you have designated control options, your TSS commands automatically propagate to the default nodes.

The following control options tailor the environment for the Command Propagation Facility:

### **CPF(ON|OFF|KILL)**

Indicates whether CPF should be activated at CA-Top Secret startup (ON|OFF) and allows you to temporarily terminate the CPF subtask (KILL) without bringing down all of CA-Top Secret.

When CPF is set to **ON**, TSS commands are transmitted by this node or received from other nodes.

When CPF is set to **OFF**, no TSS commands can be transmitted.

The **KILL** option is issued with a TSS MODIFY command to temporarily terminate the CPF subtask and automatically take a dump. The subtask can then be reattached by specifying TSS MODIFY(CPF(ON)).

### **CPFNODES(node1,node2 (S)|(R)@,...)**

Identifies the remote CA-Top Secret nodes from and/or to which CPF can propagate commands. An **(S)** indicates that the local node can only send commands to the designated remote node. An **(R)** indicates that the local node can only receive commands from the designated remote node.

**Note:** User-initiated changes (such as updated passwords or suspension due to access violations) are propagated to those nodes identified by the **CPFNODES** control option. For more information on CPF-related control options, refer to 9.2, "CPF-Related Control Options."

**CPFRVCVUND(YES|NO)**

Indicates whether or not the local node will receive commands issued from a remote node that hasn't been defined to the CPFNODES list. The default is **NO**—the local node **won't** receive commands from undefined remote nodes.

**CPFWAIT(YES|NO)**

Sets a default value for the TSS command WAIT keyword.

If CPFWAIT is omitted, CA-Top Secret chooses a default of **YES**. This means that commands are processed on a synchronous basis, requiring the user to wait for the commands to complete on **all** specified nodes before the local command completes.

If **NO** is selected, processing occurs asynchronously.

Regardless of whether you select YES or NO, the CPFWAIT control option can be overridden by the WAIT value on the individual TSS command.

**CPFTARGET(AUTO|\*|LOCAL)**

Sets a default value for the TSS command TARGET keyword.

The security administrator can select one of three options.

- **AUTO** indicates that, if a target node isn't explicitly identified on a command, that command will automatically propagate to those nodes identified by the ACID's DEFNODES. A more complete discussion of the connection between CPFTARGET and DEFNODES is included later in this chapter under the headings *Using DEFNODES on a TSS Command*, and *Administering an ACID's DEFNODES*.
- The asterisk (\*) indicates all nodes defined as send-only or send/receive in the CPFNODES control option. Nodes defined as receive-only won't be included.
- **LOCAL** indicates a particular local node.

For more details, refer to the *Control Options Guide*.

## 9.3 CA-Top Secret Command Keywords Used With CPF

The TSS command functions that can be used with CPF are authorization commands such as ADD, PERMIT, REMOVE, and CREATE as well as, WHOHAS, LIST, and WHOOWNS. The CPF keywords that can be used with these functions are:

### **TARGET(node1,node2...)**

Identifies each node to which a command can be propagated.

### **TARGET(\*)**

Transmits the command to the local node and to all nodes defined in the CPFNODES control option.

### **TARGET(=)**

Restricts command execution to the local node only. The TARGET(=) keyword overrides the CPFTARGET(\*) control option.

### **TARGET(n...n\*)**

Transmits all commands to nodes whose names begin with the indicated string. The string can range from one to seven characters.

### **DEFNODES(node1,node2,...)**

Defines default remote nodes for use in the event that a security administrator doesn't specify a TARGET keyword on a command. DEFNODES only applies if the CPFTARGET control option has been set to AUTO.

**Note:** A more complete discussion of the relationship between CPFTARGET and DEFNODES and of how an ACID's DEFNODES are administered is included in the next two sections.

### **WAIT(Yes|No)**

Sets the processing mode for the command being issued. **WAIT(YES)** selects synchronous processing. **WAIT(NO)** selects asynchronous processing. The WAIT keyword overrides the CPFWAIT control option setting.

The example below displays the users on CPU1, CPU2, and the local node that have access to payroll data.

```
TSS WHOHAS DSN(PAYROLL.) TARGET(CPU1,CPU2,=) WAIT(Y)
```

The next example grants ownership of the SYS1 data set prefix to DEPT01 on all remote R-prefixed nodes identified by the CPFNODES control option.

```
TSS ADD(DEPT01) DSN(SYS1.) TARGET(R*) WAIT(Y)
```

The final example designates the ALT, BOS, and CIN nodes as the default routing nodes for USER01.

```
TSS ADD(USER01) DEFNODES(ALT, BOS, CIN)
```

For more information on these keywords and their authorities, refer to the *Command Functions Guide*.

### 9.3.1 Using DEFNODES With a TSS Command

Although asterisk (\*) and LOCAL are the more commonly used values with CPFTARGET, when using DEFNODES with a TSS command, AUTO is required.

DEFNODES only applies under two conditions:

- If CPFTARGET is set to AUTO *and*
- No TARGET keyword is specified on the command.

If these two conditions are met, CA-Top Secret automatically retrieves the DEFNODES normally associated with the **targeted ACID**. The only time the DEFNODES keyword is actually supplied in a command is when the ACID's DEFNODES are being designated (on the initial TSS CREATE or later through a TSS ADD) or updated. DEFNODES administration is discussed in the next section.

**Note:** If CPFTARGET is set to AUTO and no DEFNODES are specified, command routing is done to all the nodes with send abilities.

If the command being issued is an ADD/REMOVE, ADMIN/DEADMIN, DELETE, MOVE, PERMIT/REVOKE, RENAME, or REPLACE, the destination nodes are taken from the DEFNODES of the **targeted ACID**.

### 9.3.2 Administering an ACID's DEFNODES

DEFNODES can be assigned to an ACID in one of three ways:

#### 1. Using TSS CREATE.

If you are using TSS CREATE, you need to specify the DEFNODES keyword. For example, the command shown below creates an ACID of USER01 for John Smith and designates NODEA, NODEB, and NODEC as his DEFNODES.

```
TSS CREATE(USER01) NAME('John Smith') TYPE(USER)
 PASSWORD(shsh,30,exp) DEFNODES(NODEA,NODEB,NODEC)
```

#### 2. Using TSS CREATE with a model ACID.

If you are using a model ACID, that ACID must have DEFNODES to be transferred to the new ACID. For example, the command shown below uses the ACID created in the previous example as the basis for Sam Jone's ACID, USER02. Since USER01 has DEFNODES, these same DEFNODES are being transferred to USER02.

```
TSS CREATE(USER02) USING(USER01) Name('Sam Jones')
```

**Note:** Model ACIDs can only be used as a basis for creating other ACIDs of the same TYPE. For example, you can't use a model ACID whose TYPE is USER to create another ACID whose TYPE will be DCA.

3. Using TSS ADD with an existing ACID.

You can add DEFNODES to an existing ACID. For example, the command shown below indicates that ACID01 now has DEFNODES of NODEB and NODEC.

```
TSS ADD(ACID01) DEFNODES(NODEB,NODEC)
```

**Note:** If a TSS CREATE is issued **without** indicating either the DEFNODES keyword, or by using a model ACID that doesn't have DEFNODES, no DEFNODES are supplied.

In addition to TSS ADD, you can also use TSS REMOVE and TSS REPLACE to update an ACID's existing DEFNODES definitions.

TSS REMOVE is used to delete one or more entries. For example, the command shown below removes NODEB from USER02's DEFNODES list while leaving NODEA and NODEC.

```
TSS REM(USER02) DEFNODES(NODEB)
```

TSS REPLACE deletes an ACID's existing DEFNODES definitions **in their entirety** and replaces them with a completely new list. For example, the command shown below completely removes the current DEFNODES from USER02 and replaces them with NODEF, NODEG, and NODEH.

```
TSS REP(USER02) DEFNODE(NODEF,NODEG,NODEH)
```

**Note:** A maximum of five DEFNODES can be added, removed, or replaced in a single TSS command.

### 9.3.3 What Happens When a Command is Issued

To get a better understanding of how the CPF control options and CPF command keywords work together, consider the consequences of the following sample commands:

**1. TSS ADD(USER01) DSN(ABC123)**

Since no TARGET was specified, CA-Top Secret looks to the CPFTARGET control option setting to determine the designated default routing procedure:

- a. If CPFTARGET is set to AUTO, the command will propagate to those nodes identified by the ACID's DEFNODES.
- b. If CPFTARGET is set to LOCAL, the command will only execute on the local node.
- c. If CPFTARGET is set to \*, the command will propagate to all those nodes identified by the CPFNODES control option.
- d. If CPFTARGET hasn't been activated, the command will only execute on the local node.

**2. TSS ADD(USER01) DSN(ABC123) TARGET(\*)**

In this case, USER01 is being granted ownership of the ABC123 data set. Since a TARGET of \* was specified, this command will be propagated to all nodes identified by the CPFNODES control option.

**3. TSS ADD(USER01) DSN(ABC123) TARGET(=)**

In this case, the same command will only be executed on the local node (as specified by the =).

**4. TSS ADD(USER01) DSN(ABC123) TARGET(NODEA,NODEB)**

Here the command will be propagated to NODEA and NODEB regardless of the values specified by the CFPNODES or CPFTARGET control options and any DEFNODES USER01 may have. An explicit TARGET will override the defaults.

## 9.4 CPF Recovery File

The CPF Recovery File is a BDAM disk file used by CPF to save transmitted commands until a response to those commands has been received from a remote machine. Only commands selecting or defaulting to WAIT(NO) are saved for retransmission. Commands targeted only for the local machine aren't saved.

When a TSS command with WAIT(NO) is entered, CPF saves a command image on the CPF Recovery File before transmitting it over the link. When a response from the remote is returned, CPF deletes the command from the file. If the response isn't received due to link failure, system down, and so forth, CPF will scan the Recovery File at the resumption of service and select all commands that were sent out but not responded to, and retransmit those commands. When a response is eventually received, the command(s) are deleted from the file.

CPF performs a scan of the CPF Recovery File at the following times:

- At CA-Top Secret initialization.
- When contact with a remote machine is reestablished after having been lost.

Before you can use the Recovery File, it must be formatted through TSSMAINT. Sample JCL is included in the *Installation and Maintenance Guide*. A DLBL statement must be inserted into the CA-Top Secret jobstream to define the CPF Recovery File. A sample DLBL statement follows.

```
// DLBL CAICPRF, 'CAI.CPF.RECOVERY', ,99/365
```

It is important to note that the **CPF Recovery File cannot be shared across multiple systems.**

If the CPF Recovery File isn't defined, command routing through CPF can still occur but there will be no retransmission of unresponded commands. Refer to the *Installation and Maintenance Guide* for more information about the CPF Recovery File.

If the CPF Recovery File becomes temporarily filled, a message is written to the job CA-Top Secret LOG and console each time CPF wants to write a message to the file but can't. The CPF operation continues but, in case of failure, the unwritten command can't be recovered.



## 9.5 Recovery and Accountability

CA-Top Secret provides recovery processing services for the Security File through the TSSRECVR utility. (No other files can be recovered using this utility.)

**TSSRECVR Utility:** When you run TSSRECVR (the CA-Top Secret utility that generates TSS commands to recreate the Security File), the TARGET is **always local** no matter what the original TARGET destination.

**TSS Command Execution:** Anyone with MISC2(TARGET) authority can enter a TSS command with a targeted destination. The command is executed at the remote machine under the authority the issuing ACID has on the remote node (regardless of his authority on the local node).

For example, ACID(HARRY) is defined as an SCA on his local machine, but on REMOTEB ACID(HARRY) is only a USER. Any command HARRY sends to REMOTEB will be executed with his authority on REMOTEB—as a user.

**Note:** If the security administrator issuing the command doesn't exist on the remote node to which the command is being propagated, you will receive the following message:

```
TSS0324E ADMINISTRATOR'S ACID DOES NOT EXIST ON TARGET NODE
```

To avoid this error, you should ensure that the administrating ACID is defined to the remote node before any commands are issued to that node.

### 9.5.1 Security Files Among Networked Machines

As stated earlier, each command routed by CPF is executed at the remote machine as though it originated there.

Resources, ACIDs, and so forth, may or may not exist on one or more remote node machines. Commands that work at one remote may fail at another; and commands executed at a remote node may not have their intended effect.

Using the previous example, if the MSCA gave ownership of DSN(SYS1) to ACID(HARRY) at his local node (where he is designated as an SCA) that command, if routed to REMOTEB through CPF, can give the same ownership to USER(HARRY).

## 9.5.2 System Entry Validation and Password Propagation

When a user is required to update his password as part of system entry validation, CA-Top Secret notifies all other CPF connected nodes of that change. If a matching ACID is found on a remote node, CA-Top Secret first compares the password of the "remote" ACID with the *old* password of the "local" ACID to verify that they aren't two different ACIDs with the same ACID name. If the passwords match, the password on the remote node is updated.

For example, USER01 signs on to TSO on Node A. His current password of "remember" has expired and he changes it to "always". Through CPF, that change is sent to Node B, a remote node. When CPF finds a USER01 in the Security File of Node B, it asks "Is the current password remember?".

- If the answer is **yes**, the password of USER01 on Node B is also updated to "always".
- If the answer is **no**, CPF assumes that the USER01 on Node B isn't the same person as the USER01 on Node A and leaves the Node B password unchanged.

When CPF detects that the passwords don't match, the following message is sent to the CPF spool data set on the remote node:

```
TSS0422E PASSWORD VERIFICATION FAILED ON REMOTE NODE
```

## 9.5.3 Installation Exit

The installation exit routine can be called for CPF transmission on both the sending and receiving side, and may block the command at either point. The exit can also make some changes to the command text. For more details, refer to 11.3, "Extending Security Through Site Security Exits" on page 11-47.

## 9.5.4 TSSCPR Utility

The TSSCPR utility is run against the CPF Recovery File to produce a flat file record. This file can then be filtered either through the TSSREPORT3 EARL report option or through another report writer to depict the contents of the CPF Recovery File.

For more information on using TSSCPR and TSSREPORT3, refer to your *Report and Tracking Guide*.

## Chapter 10. Protecting Facilities

---

**What is a Facility?:** CA-Top Secret considers a facility as a way of grouping options and associating them with a particular service that users sign on to. Examples of such services are BATCH and CICS.

Some services—such as BATCH and TSO—are automatically associated with the facility of the same name. Others—such as CICS, CA-IDMS, and DL/1—must be associated with an appropriate facility. This association happens in one of two ways:

1. Explicitly or
2. Implicitly

An **explicit** association is done by assigning a MASTFAC parameter to the ACID that is executing the batch job which created the service. An **implicit** association occurs when CA-Top Secret looks at the program in control and matches it with the program name specified in the facility.

Facilities can also be divided into two categories:

1. Single user address space (like TSO and BATCH)

The term "single user address space" comes from each signed on user getting his own address space.

2. Multiple user address space (like CICS and DL/1)

Using CICS as an example, each user signed on to a CICS region is given part of the region's block of space, resulting in the term "multiple user address space". Moreover, when a user requests access, standard CICS will only identify to MVS the Security Record for the region involved, not the specific user's Security Record. This procedure is typical of multiple user address space systems.

Refer to your facility-specific *Implementation Guide* for a more detailed description of single and multiple user address spaces.

CA-Top Secret comes equipped with a set of predefined facilities. A list of these facilities and their particular attributes (like ID and security mode) is kept in the Facilities Matrix Table. These attributes are called the facility's definition.

**Tailoring Facility Access:** To sign on to a service, a user must be given access to the facility. There are many options within the facility definition that can be used to tailor who has access to the facility and how security is to be handled for users of the service associated with that facility. These options are specified using the FACILITY control option, which is discussed in 10.1.4, "How to Change a Facility's Definition" on page 10-6.

The next sections discuss:

- How to administer the Facility Matrix Table.
- What facility-specific security CA-Top Secret provides for:
  - CICS
  - TSO
  - CA-IDMS
  - CA-Roscoe
  - DL/1
  - BATCH
  - VM

## 10.1 Administering the Facility Matrix Table

When you first install CA-Top Secret, the Facility Matrix Table that resides in the Parameter File contains a set number of predefined facilities and a number of templates (or dummy facility entries) that you can customize to create additional facilities. Each predefined and customized facility exists as an entry in this table and contains facility-specific information which controls how the facility functions under CA-Top Secret. The facility-specific information is added to the Parameter File using the FACILITY control option and its attributes. Attributes can then be changed temporarily using the TSS MODIFY command function and its parameters.

The following sections explain how the security administrator can:

- Display the contents of the Facility Matrix Table.
- Display the default values for one facility.
- Add a new facility.
- Change the default attributes for an existing facility.
- Change a facility's name.

At the conclusion of these sections is a list of the default attributes for some of the predefined facilities in the Facility Matrix Table.

### 10.1.1 How to Display the Facility Matrix Table's Contents

The definitions of all facilities currently in use at your site are contained in the Facility Matrix Table. To display them, use the TSS MODIFY command function with the ALL parameter, as shown below.

```
TSS MODIFY(FAC(ALL))
```

### 10.1.2 How to Display a Facility's Definition

To display one facility's definition, use the TSS MODIFY command function with the name of the facility as the parameter. The example shown below displays the TSO facility's definition.

```
TSS MODIFY(FAC(TSO))
```

**Sample Entry for TSO:** The default values for the TSO Facility Matrix Table entry are shown below. The first line of text in the example identifies TSO to CA-Top Secret. The remaining lines specify which FACILITY operands pertain to the TSO facility within the CA-Top Secret security environment—that is, how CA-Top Secret will treat users signed on to the TSO facility.

```
INITPGM=IKJEFLC ID=T TYPE=03
ATTRIBUTES=IN-USE,ACTIVE,SHRPRF,NOASUBM,ABEND,SUAS,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,RNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,TSOC,LCFCMD
ATTRIBUTES=MAGLC,NOTRACE,NOEODINIT,IJU,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL DOWN=GLOBAL LOGGING=ACCESS,INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
```

A list of the defaults for each facility is included in each facility's *Implementation Guide*, as well as in 10.1.6, "Default Values for Predefined Facilities" on page 10-9.

### 10.1.3 How to Add a New Facility

In addition to the pre-defined facility entries, 221 dummy facility entries are available for site customization. These predefined entries are called "dummy facilities" and are to be used for site-defined facilities, such as additional test regions for CICS or DL/1. Each of these "dummy facilities" is given the USER $nnn$  name (where  $nnn$  is a number from 0 to 221).

The following table and example show how each facility has an accompanying field ID of two characters. For example, for facility USER0 the ID= is 0 and for facility USER221 the ID= is M1.

| Facilities        | ID Field      |
|-------------------|---------------|
| USER0 - USER09    | 0 through 99  |
| USER100 - USER109 | A0 through A9 |
| USER110 - USER119 | B0 through B9 |
| USER120 - USER129 | C0 through C9 |
| USER130 - USER139 | D0 through D9 |
| USER140 - USER149 | E0 through E9 |
| USER150 - USER159 | F0 through F9 |
| USER160 - USER169 | G0 through G9 |
| USER170 - USER179 | H0 through H9 |
| USER180 - USER189 | I0 through I9 |
| USER190 - USER199 | J0 through J9 |
| USER200 - USER209 | K0 through K9 |
| USER210 - USER219 | L0 through L9 |
| USER220 - USER221 | M0 through M1 |

### USERnnn

```
INITPGM=***** ID=xx TYPE=99
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,RNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC,LCFTRANS
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
```

To add a new facility, you must modify one of the dummy facility entries. The entry can be changed either temporarily (with TSS MODIFY) or permanently (with the FACILITY control option). Using one of these methods, change the attributes listed below.

#### INITPGM=

Replace the asterisks with the name of the initialization program.

#### ID=

Replace nn with the number in the USERnn name.

#### TYPE=

If you are creating a duplicate facility for CICS, DL/1, or CA-IDMS, replace the current value with either CICS, DL/1, or IDMS respectively.

**Tip:** When creating a duplicate facility, use the existing entry as a model.

### 10.1.4 How to Change a Facility's Definition

Values contained in the Facility Matrix Table can be changed temporarily or permanently. To change values temporarily—for the current session of CA-Top Secret—use the syntax of the TSS MODIFY command function shown below.

```
TSS MODIFY(FAC(suboption=operand=value))
```

**suboption** The name of the facility whose defaults are being changed.

**operand** An operand of the FACILITY control option (like MODE)

**value** A value for the operand, if it has one (like WARN for MODE)

**Changing Values Permanently:** To change a value permanently, use the syntax of the FACILITY control option shown below.

```
FACILITY(facility-name=suboption=value,suboption=value,...)
```

If the string of suboptions and values won't fit on one line, you can enter the option and suboptions on multiple lines as shown next.

```
FAC(TSO=MODE=IMPL,LOG=(ALL))
FAC(TSO=LOCKTIME=5)
FAC(TSO=NOLUMSG,RNDPW)
```

**Note:** Facility suboptions override global control options of the same name.

When entered into the Parameter File using the FACILITY control option, facility defaults can be overridden using the TSS MODIFY command function. However, defaults revert to the ones listed in the Parameter File when CA-Top Secret is restarted after a shutdown.

**FACILITY Operands:** The following list of terms describes the FACILITY control option operands listed in the TSO example on 10-4. There are additional operands—most notably for CICS. For a complete list of FACILITY control option operands, refer to your *Control Options Guide*.



**Temporary Change Examples:** To temporarily change the mode value for the BATCH facility from FAIL to WARN, enter:

```
TSS MODIFY(FACILITY(BATCH=MODE=WARN))
```

To temporarily prevent users from signing on to the CICSTEST facility, enter:

```
TSS MODIFY(FAC(CICSTEST=INACTIVE))
```

**Permanent Change Examples:** To permanently change the mode value for the BATCH facility from FAIL to WARN, enter the following command in the CA-Top Secret Parameter file:

```
FAC(BATCH=MODE=WARN)
```

To permanently prevent user type ACIDs from receiving last used messages at sign on, enter:

```
FAC(CICSTEST=NOLUMSG)
```

## 10.1.5 How to Change a Facility's Name

To change a facility's name permanently (the recommended method), use the FACILITY control option and the NAME suboption. The following example changes the name of the USER4 facility to CICSА.

```
FAC(USER4=NAME=CICSA)
```

If the name of a facility is changed, then all subsequent control option entries that impact the newly-named facility must contain the new facility name. For example, to change the name of USER4 to CICSА and force CICSА to process in WARN mode, enter:

```
FAC(USER4=NAME=CICSA)
FAC(CICSA=MODE=WARN)
```

On the other hand, the following entry will **not** affect the mode of the newly-named facility. It will produce an error message because the referenced facility no longer exists.

```
FAC(USER4=NAME=CICUSER)
FAC(USER4=MODE=WARN)
```

Control option changes made before a facility is renamed remain in effect for the newly named facility. For example, if these entries are made:

```
FAC(USER4=MODE=WARN)
FAC(USER4=NOABEND)
FAC(USER4=NAME=CICSA)
```

the newly-named CICSА facility will process security in WARN mode and will have the NOABEND attribute. Of course, any additional changes must be made to CICSА, not USER4.

## 10.1.6 Default Values for Predefined Facilities

This section lists the default attributes of the predefined facilities in the Facility Matrix Table.

### ACEP

```
INITPGM=ACE ID=A TYPE=27
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC,LCFCMD
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
MAXUSER=03000 PRFT=003
```

### APPC:

```
INITPGM=ATB ID=AP TYPE=03
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=IN-USE,ACTIVE,NOSHRPRF,NOASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC,LCFCMD
ATTRIBUTES=MSGLC,NOTRACE,EODINIT,DORMPW,NONPWR,NOIMSXTND
MODE=FAIL DOWN=GLOBAL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
MAXUSER=03000 PRFT=003
```

### BATCH:

```
INITPGM=IEFIIC ID=B TYPE=01
ATTRIBUTES=IN-USE,ACTIVE,SHRPRF,NOASUBM,ABEND,SUAS,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,NOWARNPW,NOTSOC,LCFCMD
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9,SMF
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
```

### CA7:

```
INITPGM=UCC ID=U TYPE=25
ATTRIBUTES=ACTIVE,SHRPRF,NOASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=NOLUMSG,NOSTMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC,LCFCMD
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
MAXUSER=03000 PRFT=003
```

**CICSPROD**

```

INITPGM=DFH ID=C TYPE=04
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,RNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,NORES,WARNPW,NOTSOC,LCFTRANS
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,SMF,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
FACMATRX=NO EXTSEC=YES
XJCT=YES XFCT=YES XCMD=YES XDCT=YES XTRAN=YES
XTST=YES XPSB=YES XPCT=YES XPPT=YES XAPPC=NO
PCTEXTSEC=OVERRIDE PCTCMDSEC=OVERRIDE
DSNCHECK=NO LTLOGOFF=NO RLP=NO SLP=NO
MAXSIGN=050,KILL MAXUSER=03000

```

```

BYPASS: RESOURCE=TRANID NAMES: CAQP CATD CATP
 CAUT CBRC CCMF CECS CESF CESN
 CLS1 CLS2 CMSG CMPX CNSE CNPX
 CRDR CQRY CRSQ CRSR CRSY CRSR
 CRTE CRTR CSAC CSCY CSGM CSGX
 CSFU CSIR CSJC CSKP CSLG CSMI
 CSM1 CSM2 CSM3 CSM4 CSM5 CSNC
 CSPG CSPK CSRK CSPP CSPQ CSPS
 CSRS CSSC CSSF CSSN CSSX CSSY
 CSTA CSTB CSTE CSTT CSXX CVST
 CWTR CWTO CXCU CXRE TSLK TSLO
 TSS TSSC 8888 9999 CEDF
 CLS3
 CXRT CLS4

```

**CICSTEST**

```

INITPGM=DFH ID=K TYPE=04
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,RNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,NORES,WARNPW,NOTSOC,LCFTRANS
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,SMF,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
FACMATRX=NO EXTSEC=YES
XJCT=YES XFCT=YES XCMD=YES XDCT=YES XTRAN=YES
XTST=YES XPSB=YES XPCT=YES XPPT=YES XAPPC=NO
PCTEXTSEC=OVERRIDE PCTCMDSEC=OVERRIDE
DSNCHECK=NO LTLOGOFF=NO RLP=NO SLP=NO
MAXSIGN=050,KILL MAXUSER=3000

```

```

BYPASS: RESOURCE=TRANID NAMES: CAQP CATD CATP
 CAUT CBRC CCMF CECS CESF CESN
 CRDR CQRY CRSQ CRSR CRSY CRSR
 CRTE CRTR CSAC CSCY CSGM CSGX
 CSFU CSIR CSJC CSKP CSLG CSMI
 CSM1 CSM2 CSM3 CSM4 CSM5 CSNC
 CSPG CSPK CSRK CSPP CSPQ CSPS
 CSRS CSSC CSSF CSSN CSSX CSSY
 CSTA CSTB CSTE CSTT CSXX CVST
 CWTR CWTO CXCU CXRE TSSS TSLO
 TSLA TSLM 8888 9999 CLS3 CXRT

```

**COMPLETE:**

```

INITPGM=THR ID=C TYPE=21
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC,LCFCMD
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8

```

**CONSOLE:**

```

INITPGM=*** ID=CN TYPE=02
ATTRIBUTES=ACTIVE,NOSHRPRF,NOASUBM,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC,LCFCMD
ATTRIBUTES=MSGLC,NOTRACE,EODINIT,DORMPW,NONPWR,
MODE=FAIL DOWN=BYPASS LOGGING=ACCESS,INIT,SMF,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
MAXUSER=03000 PRFT=003

```

**DB2PROD**

INITPGM=CAD ID=DB TYPE=00  
ATTRIBUTES=ACTIVE, SHRPRF, ASUBM, NOABEND, MULTIUSER, NOXDEF  
ATTRIBUTES=LUMSG, STMSG, SIGN(M), INSTDATA, RNDPW, AUTHINIT  
ATTRIBUTES=NOPROMPT, NOAUDIT, NORES, WARNPW, NOTSOC, LCFTRANS  
ATTRIBUTES=MSGLC, NOTRACE, NODORMPW, NONPWR, NOIMSXTND  
MODE=FAIL LOGGING=INIT, MSG, SEC9  
UIDACID=8 LOCKTIME=000 DEFACID=\*NONE\* KEY=8

**DB2TEST:**

INITPGM=CAD ID=DT TYPE=00  
ATTRIBUTES=ACTIVE, SHRPRF, ASUBM, NOABEND, MULTIUSER, NOXDEF  
ATTRIBUTES=LUMSG, STMSG, SIGN(M), INSTDATA, RNDPW, AUTHINIT  
ATTRIBUTES=NOPROMPT, NOAUDIT, NORES, WARNPW, NOTSOC, LCFTRANS  
ATTRIBUTES=MSGLC, NOTRACE, NODORMPW, NONPWR, NOIMSXTND  
MODE=FAIL LOGGING=INIT, MSG, SEC9  
UIDACID=8 LOCKTIME=000 DEFACID=\*NONE\* KEY=8

**ENVIRON:**

INITPGM=ENV ID=E TYPE=15  
ATTRIBUTES=ACTIVE, SHRPRF, NOASUBM, ABEND, MULTIUSER, NOXDEF  
ATTRIBUTES=LUMSG, STMSG, SIGN(M), INSTDATA, NORNDPW, AUTHINIT  
ATTRIBUTES=NOPROMPT, NOAUDIT, NORES, WARNPW, NOTSOC, LCFCMD  
ATTRIBUTES=MSGLC, NOTRACE, NODORMPW, NONPWR, NOIMSXTND  
MODE=FAIL  
LOGGING=INIT, MSG, SEC9  
UIDACID=8 LOCKTIME=000 DEFACID=\*NONE\* KEY=8

**HSM:**

INITPGM=ARC ID=H TYPE=099  
ATTRIBUTES=IN-USE, ACTIVE, SHRPRF, NOABEND, SUAS, NOXDEF  
ATTRIBUTES=NOASUBM, MSGLC, NOEODINIT, IJU  
ATTRIBUTES=NOLUMSG, NOSTMSG, SIGN(M), INSTDATA, NORNDPW, AUTHINIT  
ATTRIBUTES=NOPROMPT, NOAUDIT, RES, NOWARNPW, NOTSOC, LCFCMD  
ATTRIBUTES=NOTRACE, NODORMPW, NONPWR, NOIMSXTND  
MODE=WARN DOWN=GLOBAL LOGGING=INIT, SMF, MSG, ACCESS, SEC9  
UIDACID=8 LOCKTIME=000 DEFACID=\*NONE\* KEY=8

**IDMSPROD:**

INITPGM=RHD ID=M TYPE=11  
ATTRIBUTES=ACTIVE, SHRPRF, ASUBM, NOABEND, MULTIUSER, NOXDEF  
ATTRIBUTES=LUMSG, STMSG, SIGN(M), INSTDATA, RNDPW, AUTHINIT  
ATTRIBUTES=NOPROMPT, NOAUDIT, NORES, WARNPW, NOTSOC, LCFTRANS  
ATTRIBUTES=MSGLC, NOTRACE, NODORMPW, NONPWR, NOIMSXTND  
MODE=FAIL LOGGING=ACCESS, INIT, MSG, SEC9  
UIDACID=8 LOCKTIME=000 DEFACID=\*NONE\* KEY=8

**IDMSTEST**

```

INITPGM=RHD ID=Q TYPE=11
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,RNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,NORES,WARNPW,NOTSOC,LCFTRANS
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8

```

**IMSPROD:**

```

INITPGM=DFS ID=I TYPE=05
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,RNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,NORES,WARNPW,NOTSOC,LCFTRANS
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8

```

**IMSTEST:**

```

INITPGM=DFS ID=X TYPE=05
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,RNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,NORES,WARNPW,NOTSOC,LCFTRANS
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8

```

**INTERACT:**

```

INITPGM=MEN ID=I TYPE=14
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC,LCFCMD
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=5

```

**JES:**

```

INITPGM=HAS ID=J TYPE=12
ATTRIBUTES=ACTIVE,NOSHRPRF,NOASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC,LCFCMD
ATTRIBUTES=MSGLC,NOTRACE,DORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8

```

**NCCF**

INITPGM=DSI ID=N TYPE=06  
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,ABEND,MULTIUSER,NOXDEF  
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,NORNDPW,NOAUTHINIT  
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC,LCFCMD  
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND,NOEODINIT,IJU  
MAXUSER=03000, PRFT=003 LOGGING=INIT,MSG DOWN=GLOBAL  
UIDACID=8 LOCKTIME=000 DEFACID=\*NONE\* KEY=8

**ROSCOE:**

INITPGM=ROS ID=R TYPE=07  
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,NOABEND,MULTIUSER,NOXDEF  
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,RNDPW,AUTHINIT  
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC,LCFCMD  
ATTRIBUTES=NOTRACE,NODORMPW,NONPWR,NOIMSXTND,MSGLC  
MODE=FAIL LOGGING=INIT,MSG,SEC9  
UIDACID=8 LOCKTIME=000 DEFACID=\*NONE\* KEY=8

**STC:**

INITPGM=IEESB605 ID=S TYPE=02  
ATTRIBUTES=IN-USE,ACTIVE,SHRPRF,NOASUBM,ABEND,SUAS,NOXDEF  
ATTRIBUTES=LUMSG,NOSTMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT  
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,NOWARNPW,NOTSOC,LCFCMD  
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND  
MODE=FAIL LOGGING=INIT,MSG,SEC9  
UIDACID=8 LOCKTIME=000 DEFACID=\*NONE\* KEY=8

**STONE:**

INITPGM=TON ID=T TYPE=13  
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,ABEND,MULTIUSER,NOXDEF  
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,RNDPW,AUTHINIT  
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,TSOC,LCFCMD  
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND  
MODE=FAIL LOGGING=ACCESS,INIT,MSG,SEC9  
UIDACID=8 LOCKTIME=000 DEFACID=\*NONE\* KEY=8

**TSO:**

INITPGM=IKJEFLC ID=T TYPE=03  
ATTRIBUTES=IN-USE,ACTIVE,SHRPRF,NOASUBM,ABEND,SUAS,NOXDEF  
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,RNDPW,AUTHINIT  
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,TSOC,LCFCMD  
ATTRIBUTES=NOTRACE,NODORMPW,NONPWR,NOIMSXTND,MSGLC  
MODE=FAIL LOGGING=INIT,MSG,SEC9  
UIDACID=8 LOCKTIME=000 DEFACID=\*NONE\* KEY=8



There are 221 "dummy" facility definitions you can modify for use at your site. All default values for these definitions are the same with the exception of ID=. How to modify these entries is covered in 10.1.3, "How to Add a New Facility" on page 10-4 and how to display definitions is covered in 10.1.2, "How to Display a Facility's Definition" on page 10-4.

### USER0

```
INITPGM=000 ID=A TYPE=19
ATTRIBUTES=ACTIVE,SHRPRF,NOASUBM,ABEND,SUAS,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT,
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC
ATTRIBUTES=NOTRACE,NODORMPW,NONPWR,NOIMSXTND,MSGLC
MODE=FAIL LOGGING=INIT,SMG,MSG
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
```

### Through

#### USER221:

```
INITPGM=xxx ID=M1 TYPE=99
ATTRIBUTES=ACTIVE,SHRPRF,NOASUBM,ABEND,SUAS,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT,
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,NOWARNPW,NOTSOC
ATTRIBUTES=NOTRACE,NODORMPW,NONPWR,NOIMSXTND,MSGLC
MODE=FAIL LOGGING=INIT,SMF,MSG
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
```

#### VAMSPF:

```
INITPGM=VAM ID=V TYPE=09
ATTRIBUTES=ACTIVE,SHRPRF,NOASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,RNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,TSOC,LCFCMD
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
```

#### VM:

```
INITPGM=TSS ID=V TYPE=08
ATTRIBUTES=ACTIVE,SHRPRF,NOASUBM,ABEND,SUAS,NOXDEF
ATTRIBUTES=NOLUMSG,NOSTMSG,SIGN(M),INSTDATA,RNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC,LCFCMD
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=*NONE* KEY=8
```

**WYLBUR**

```
INITPGM=UEX ID=W TYPE=10
ATTRIBUTES=ACTIVE,SHRPRF,ASUBM,NOABEND,MULTIUSER,NOXDEF
ATTRIBUTES=LUMSG,STMSG,SIGN(M),INSTDATA,NORNDPW,AUTHINIT
ATTRIBUTES=NOPROMPT,NOAUDIT,RES,WARNPW,NOTSOC,LCFCMD
ATTRIBUTES=MSGLC,NOTRACE,NODORMPW,NONPWR,NOIMSXTND
MODE=FAIL LOGGING=INIT,MSG,SEC9
UIDACID=8 LOCKTIME=000 DEFACID=**NONE* KEY=8
```

## 10.2 Securing CICS

CA-Top Secret offers extensive and flexible security protection for CICS regions, transactions, and resources without modifying CICS code. CA-Top Secret Release 3.0 supports CICS/TS Release 1.1 and below. Multi-Region Option (MRO) and Inter-System Communications (ISC) are fully supported.

Many of the standard CA-Top Secret security features can be applied to CICS. The most useful include:

- Signon security and authorization restrictions
- Password validation
- Terminal security
- Program security
- Job submission validation
- Security administration
- Security key
- Multiple signons

### 10.2.1 Signon Security and Authorization Restrictions

To sign on to CICS, a user's ACID must be authorized to access the CICS facility. Access authorization is granted by adding the FACILITY parameter to the ACID as shown below, or by adding the facility to the ALL Record.

```
TSS ADD(USER01) FAC(CICSTEST)
```

Access to the facility can be restricted by the day of the week, the time of the day, and for a designated length of time. In addition, access can be limited to a particular terminal. All of these methods are discussed in Chapter 5, *Password and System Entry Security*.

CA-Top Secret supports both the CESN and CSSN CICS signon procedures. Both procedures require the user to enter an ACID name, password, and optional new password. Refer to *Implementation: CICS Guide* for a detailed discussion of this procedure.

CA-Top Secret signon processing will also operate correctly if your installation uses a non-standard signon procedure. However, the CICS signon program, DFHSNP, must be invoked by that procedure.

### 10.2.2 Password Validation

All of the standard password security options discussed in Chapter 5, *Password and System Entry Security*, can be used with CICS.

### 10.2.3 Terminal Security

CA-Top Secret can be used to restrict terminal use to only authorized users. Users can also be restricted to signing on only from specific terminals or accessing only particular CICS regions from certain terminals. In addition, you can prevent a user from signing on to the same region from multiple terminals.

Unattended terminals can be protected against unauthorized access by using automatic terminal locking. Moreover, cumulative security violation thresholds can be established that will automatically force terminal locking if this threshold is exceeded.

Terminal security is discussed in Chapter 5, *Password and System Entry Security*.

### 10.2.4 Program Security

CA-Top Secret provides a utility that allows security administrators to monitor security-related events for one or more systems in a realtime manner. This utility can be executed under CICS and is called TSSTRACK. For additional information on TSSTRACK, refer to your *Report and Tracking Guide*.

### 10.2.5 Job Submission Validation

Online jobs submitted under CICS are treated by CA-Top Secret exactly as if they were submitted under VM.

### 10.2.6 Security Administration

Security administrators can use the TSS command under CICS to perform all security administration. The syntax of the command is discussed in 1.2.5.2, "What are Command Functions?" on page 1-16.

Changes to the security database made through a TSS command are immediately recognized by all facilities. As a result, a user's BATCH access could be administered during a CICS terminal session. However, any changes made to an ACID's security record while that ACID is signed on do not immediately take affect. The ACID must sign off and then sign back on for the changes to be registered.

## 10.2.7 Security Key

A user's CICS security key can be specified using the SCTYKEY keyword on the TSS CREATE or ADD command function. Up to 64 security keys can be specified. Security keys are available in all modes. For a detailed description of the SCTYKEY keyword, refer to the *Command Functions Guide*.

## 10.2.8 Multiple Signons

CA-Top Secret security for CICS supports multiple concurrent signons by an ACID. This is a site-selectable option.

Use the SIGN(M) control option to allow simultaneous logons for the same ACID. Use the SIGN(S) control option to disallow simultaneous logons for the same ACID for each CICS region running under a specified facility.

## 10.3 Securing CA-IDMS

CA-Top Secret provides comprehensive security capabilities that work with CA-IDMS security to effectively extend resource control within the CA-IDMS environment. The key areas in which CA-Top Secret provides protection for CA-IDMS are:

- Signon security and authorization restrictions
- Transaction security
- Password validation
- Program and subschema security
- Area security
- Terminal security
- Job submission validation

**Security Under Release 12.0:** The CA-Top Secret CA-IDMS Interface for Release 12.0 is invoked automatically using the CA-CIS CAISSF component when external security is specified for the CA-IDMS Signon Resource Type Table (SRTT). CA-Top Secret provides security for all CA-IDMS resources that are coded in the SRTT.

For more information on providing security for CA-IDMS, refer to your CA-Top Secret/MVS *Implementation: Other Interfaces Guide* and your CA-IDMS *security administrator's Guide*.

### 10.3.1 Signon Security and Authorization Restrictions

To sign on to CA-IDMS a user's ACID must be authorized to access the CA-IDMS facility. Access authorization is granted by adding the FACILITY parameter to the ACID as shown below, or by adding the facility to the ALL Record.

```
TSS ADD(USER01) FAC(IDMSTEST)
```

Access to the facility can be restricted by the day of the week, the time of the day, and for a designated length of time. In addition, access can be limited to a particular CPU, terminal, reader, or node. All of these methods are discussed in Chapter 5, *Password and System Entry Security*.

In addition to being defined to CA-Top Secret, all users must be defined to CA-IDMS in the CA-IDMS Data Dictionary. Also, if a user is defined in the CA-IDMS Data Dictionary as having a password, then the user will be prompted twice for a password when he attempts to sign on—once by CA-IDMS for the password in the Data Dictionary, and once by CA-Top Secret for the CA-Top Secret password.

**Recommendation:** It is strongly recommended that users be defined in the CA-IDMS Data Dictionary as not having a password, thus allowing CA-Top Secret to perform **all** password checking.

### 10.3.2 Transaction Security

CA-Top Secret secures CA-IDMS transactions in two ways:

1. By protecting them at the resource level as owned resources using the OTRAN resource class.
2. On a user-by-facility basis through the Limited Command Facility (LCF).

Considerations for administering LCF and OTRAN are discussed in Chapter 7, *Resource Protection*, and in the *Implementation: Other Interfaces Guide*.

### 10.3.3 Password Validation

All of the standard password security options discussed in Chapter 5, *Password and System Entry Security*, can be used with CA-IDMS.

### 10.3.4 Program and Subschema Security

As with any resource or attribute, ownership of a program or a subschema immediately protects the resource across all facilities and regions. Once owned, access to the program and subschema must be granted using the TSS PERMIT command, and limited to only specific regions through the FACILITY keyword as part of the program and subschema permission.

Program security is discussed in Chapter 7, *Protecting Resources*, while subschema security is discussed in *Implementation: Other Interfaces Guide*.

### 10.3.5 Area Security

Area security is provided for both logical and physical databases. Ownership of an area protects the resource across all defined CA-IDMS regions. Access is then granted using the TSS PERMIT command. Use of the area can be limited to specific regions through the FACILITY keyword as part of the area definitions.

Area security is discussed in *Implementation: Other Interfaces Guide*.

### 10.3.6 Terminal Security

Terminals are owned resources, making auditing and distributed administration easily performed. Ownership of a terminal protects it across all defined facilities; however, access can be limited to only specific facilities through the FACILITY keyword as part of the terminal definition.

Terminal security is discussed in Chapter 5, *Password and System Entry Security*.

### 10.3.7 Security Administration

Security administrators can use the TSS command under CA-IDMS to perform all security administration. The syntax of the command is discussed in 1.2.5.2, "What are Command Functions?" on page 1-16.

Changes to the security database made through a TSS command are immediately recognized by all facilities. As a result, a user's access could be administered during a CA-IDMS terminal session.



## 10.4 Securing BATCH Jobs

You can use CA-Top Secret to provide security protection for the BATCH facility. The key areas in which CA-Top Secret protects the BATCH facility are:

- Signon security and authorization restrictions
- Password validation
- Card and remote reader security
- Job submission validation
- Job scheduling security

### 10.4.1 Signon Security and Authorization Restrictions

CA-Top Secret views BATCH as a facility that must be protected and authorized for use. To provide this protection, each batch job must be associated with an ACID and password. Access authorization is granted by adding the FACILITY parameter to the ACID as shown below, or by adding the facility to the ALL Record.

```
TSS ADD(USER01) FAC(BATCH)
```

Access to the facility can be restricted by the day of the week, the time of the day, and for a designated length of time. In addition, access can be limited to a particular CPU, reader, or node. All of these methods are discussed in Chapter 5, *Password and System Entry Security*.

**ACID Derivation:** CA-Top Secret treats a batch job exactly like an ACID, so that it has an associated user record with a set of specific access authorization. CA-Top Secret derives an ACID for batch jobs submitted through an online facility based on the SUBACID control option. The value for this option that is used the most often derives the ACID from the user ACID that is signed on to the online facility. This ACID derivation allows the batch job to run with the same ACID as the ACID of the online user.

Another way to derive an ACID and minimize required JCL revisions at the same time is to use the JOBACID control option. This control option derives an ACID from information on the existing JOB statement as follows:

1. If you want each job to have a unique ACID, set the JOBACID control option to derive the ACID from the jobname or programmer name on the JOB statement.
2. If you want each job to have a group ACID, set JOBACID to derive the ACID from the accounting field or the USER= field on the JOB statement.

This method also allows online user to submit batch jobs that will run under an ACID other than the ACID of the online user—called a secondary ACID. The user can code the secondary ACID in the USER= field on the JOB statement. If the online user is permitted to the secondary ACID, the user won't have to know the password for the secondary ACID. This password will be supplied in a non-viewable field by CA-Top Secret

3. If you want to control access based on the source of the job, set JOBACID to derive an ACID from the reader name.

**Note:** The derived ACID must be a valid ACID. If it isn't, the default ACID specified using the DEFACID suboption of the FACILITY control option will be applied to the job.

A description of the methods used to identify the ACID associated with a batch job can be found in the *Implementation: BATCH, STC, and APPC Guide*.

### 10.4.2 Password Validation

In IMPLEMENT and FAIL mode, a user must supply a valid password. In WARN and DORMANT mode, you can use the FACILITY control option to force the user to supply a valid password. The requirements for password validation for different modes are discussed in the *Implementation: BATCH, STC, and APPC Guide*.

### 10.4.3 Card and Remote Reader Security

For jobs being submitted from a physical reader, RJE, or NJE terminals, the submitter's password must be manually coded in the PWD= parameter on the ID statement—unless the associated ACID doesn't require a password.

### 10.4.4 Job Submission Validation

By default, CA-Top Secret allows a defined user to submit batch jobs for which the ACID is specifically authorized. Explicit authority is required to permit a user to submit jobs identified by other ACIDs. The required authority needed to submit these jobs is granted using the ACID keyword of the TSS PERMIT command function.

### 10.4.5 Job Scheduling Security

Production job scheduling systems (such as CA-Scheduler) can be given full authority to submit any job as required. This authority can be:

- limited to a specified list of jobs.
- restricted by facility.
- restricted by privileged program.

Refer to the product's documentation for more details on job scheduling security.

## 10.5 Securing ICCF

This information only applies to running batch jobs in ICCF pseudo partitions.

CA-Top Secret protects VSE libraries, files and program loads for jobs executing in ICCF pseudo partitions in the same way it supports any other VSE batch process.

Any jobs executing in an ICCF pseudo partition will execute under the facility name of the CICS region instead of the default BATCH facility.

To implement and support ICCF security checks, the RES facility option must be added for the facility in which ICCF resides:

```
TSS MODIFY(CICSPROD=RES)
```

Alternatively, it can be added permanently in the TSSPARM file.

## 10.6 Securing VM

The information in this section only applies to sites running CA-Top Secret VM.

Virtual machines running CA-Top Secret are under the VM facility. CA-Top Secret controls access to the VM facility by requiring that the user be authorized to use the virtual machine. By default, only the MSCA is authorized to use VM when CA-Top Secret is first installed. Everyone else must be explicitly authorized to use the VM facility through a TSS CREATE or TSS ADD command function. A sample command appears below.

```
TSS ADD(USER01) FAC(VM)
```

If you want to segregate your VM CPUs into different facilities, use the FACILITY control option to rename one of the USER $nm$  entries in the Facility Matrix Table. For example, the following command designates the USER1 entry as VMTEST.

```
FACILITY(USER1=NAME=VMTEST)
```

The following example sets the mode for this facility to WARN.

```
FACILITY(VMTEST=MODE=WARN)
```

Next, use the VMFAC control option to associate your CA-Top Secret facility to the DMKSYSID of the CPU by entering:

```
VMFAC(SYSTEMC=VMTEST)
```

In this example, SYSTEMC identifies the SYSID for DMKSYSID.

For more information, see the *Control Options Guide*.

## 10.6.1 How to Activate/Deactivate the VM Facility

Security administrators with the proper authority can activate or deactivate the VM facility by using either the TSS MODIFY command (temporarily) or the FACILITY control option (permanently).

To activate the facility, use the ACTIVE suboption of FACILITY as shown below, to allow users to sign on to the VMTEST facility.

```
TSS MODIFY FAC(VMTEST=ACTIVE)
```

To deactivate the facility, specify the INACT (inactive) suboption as shown below.

```
TSS MODIFY FAC(VMTEST=INACT)
```

## 10.6.2 Sharing Security Files Between VSE, VM and MVS

CA-Top Secret provides the capability to share Security Files between its VSE, MVS and VM products. This feature is optional and can be implemented at any time.

The advantages of sharing your Security File include:

- more efficient control of your security environment.
- the flexibility to administer security from the VSE, MVS or VM system.
- The ability to maintain one Security Record for each ACID, making it easier to gather user information.
- The ability to gather information about resources that are accessible from VSE, MVS and VM.

**How to Share Security Files:** The following requirements must be met to ensure that this feature functions successfully:

- Releases for VSE, MVS and/or VM must be compatible (for example, CA-Top Secret VSE 3.0 can share Security Files with CA-Top Secret VM 1.4, but can't with CA-Top Secret VM 1.1 and below).
- Encryption keys provided at installation must be identical. See the *Installation Guides* for CA-Top Secret VSE, MVS or VM for information on the encryption key.
- The SECFILE control option for CA-Top Secret VM must have a data set name that matches the data set specified on the CAISOCK DLBL statement for the CA-Top Secret VSE startup procedure.

- The SHRFILE control option for CA-Top Secret VSE, MVS and VM must be set to YES or SECURITY, so that the physical lock record (located in the Security and/or Audit files) is accessible to all systems sharing the Security File.

**Note:** VSE, MVS and VM share one Security File, but each has a separate Parameter File which stores these control options.

- In the VM directory for the CA-Top Secret server machine you must establish a multi-write link to the VSE or MVS volume that holds the Security File.

**Note:** CA-Top Secret VSE, VM or MVS don't use hardware reserve/release.





# Chapter 11. Customizing and Extending Security

---

CA-Top Secret offers several methods of customization. Before committing your installation to a particular type of customization, refer to the *Planning Guide*. This guide compares all customization techniques and discusses the advantages and disadvantages of each.

This chapter explains how to use these customization methods:

- the Application Interface
- the VSE Security Interface
- customized security exits

## 11.1 Extending Security Using the Application Interface

The Application Interface, provided with CA-Top Secret, lets a programmer easily link an application program to CA-Top Secret security services. The Application Interface can be used to:

- Perform security checks for resources, such as database fields and user resources (UR1 and UR2).
- Perform security checks on application program menu options, CICS application panels, and so on.
- Consolidate additional security and integrity requirements (for example, application level security requirements).
- Retrieve or update user-oriented information maintained on the Security Record's installation data area.
- Log site-oriented data to the Audit/Tracking File.
- Retrieve a user's ACID and status.
- Retrieve a user's list of accessible general resources.

The CA-Top Secret Application Interface can be invoked by macro or command level programs written in COBOL, PL/1, or Assembler. This interface relieves the programmer from the task of locating the user's Security Record and interfacing directly with CA-Top Secret through the VSE Security Interface.

**Note:** The Application Interface can't be used to retrieve distributed information from non-local security files that are using CPF.

### 11.1.1 How to Use the Application Interface

To use the Application Interface, the following components are required:

- the Application Interface program called TSSAI.

TSSAI is a CA-Top Secret-supplied program. It resides, and is distributed in, the CA-Top Secret load library.

**Note:** You must link to the TSSAI program dynamically.

- the appropriate Request Parameter List.

CA provides mapping through the Request Parameter List. Member TSSOPMAT, on the distribution tape, provides a parameter list for each of the languages in the following members:

| Member Name | Programming Language |
|-------------|----------------------|
| TSSCPLA     | Assembler            |
| TSSCPLC     | COBOL                |
| TSSCPLP     | PL/I                 |

**Note:** These members contain two length fields: TSSPLEN and TSSPLEN2.

TSSPLEN2 is required if you will be using FACLIST; otherwise choose TSSPLEN to reduce overhead.

The application program passes a parameter list to the program TSSAI through an appropriate Dynamic Call Mechanism in the language being used.

To use the CA-Top Secret Application Interface, the application programmer must define a parameter list declaring all of the Request Record fields. The Request Record contains two types of fields: the Request Fields, filled in by the application program, and the Return Fields, filled in by CA-Top Secret. The application program supplies information to CA-Top Secret through the Request fields. Depending on the type of checking that is to be performed, all or some of the fields are required.

**Application Program Code Logic:** To correctly invoke the Application Interface, the program code (written in COBOL, PL/I, or Assembler) must follow the logic outlined below.

1. Build the parameter list to be passed. All fields not supplied for your request should be initialized to binary zeros—except for TSSRNAME, which should be filled with blanks.
2. Invoke the CA-Top Secret Application Interface to process the request.

The CA-Top Secret Application Interface can be invoked by issuing a LINK (or a dynamic call). The examples later in this chapter apply to BATCH. Refer to the optional materials file provided on the CA-Top Secret distribution tape (file name TSSOPMAT) or to each product's *Installation Guide* for examples that apply to CIC or CA-IDMS.

3. Perform processing based on the returned data.

### 11.1.1.1 Request Record Field Characteristics

The Request Record fields and their characteristics are listed in the following table.

| Table 11-1. Request Fields |            |            |        |                                                                                                                            |
|----------------------------|------------|------------|--------|----------------------------------------------------------------------------------------------------------------------------|
| Field                      | Dec Offset | Hex Offset | Length | Content                                                                                                                    |
| TSSHEAD                    | +0         | +0         | 8      | TSSHEAD must have value TCPLV4L4                                                                                           |
| TSSCLASS                   | +8         | +8         | 8      | Class Name                                                                                                                 |
| TSSRNAME                   | +16        | +10        | 44     | Resource Name or, when extracting or updating an ACID's INSTALLATION DATA, the name of the ACID                            |
| TSSPPGM                    | +60        | +3C        | 8      | The name of the program that must be used to access the resource (privileged program name)                                 |
| TSSACC                     | +68        | +44        | 8      | Requested access level                                                                                                     |
| TSSCACEE                   | +84        | +54        | 4      | Each user's ACEE, if checked for another ACID                                                                              |
| TSSVOL                     | +88        | +58        | 6      | VOLSER used for data set requests only                                                                                     |
| TSSLOG                     | +94        | +5E        | 1      | Y logs the event and does violation logging<br>N, no logging is done; use when violation VTHRESH processing isn't required |
| TSSRTN                     | +114       | +72        | 256    | New/updated installation data                                                                                              |

**Note:** The content of the TSSHEAD field indicates the format of the request record. CA-Top Secret Release 5.0 fully supports previous formats of the parameter lists—indicated by TCPLV4L2, TCPLV4L3, and TCPLV4L4.

If the value of TSSHEAD doesn't match any supported format, TSSAI assumes that the parameter format for Release 4.1 is being used.

### 11.1.1.2 CA-Top Secret Return Data Fields

**Installation Data Return Fields:** After the application program's request is processed, CA-Top Secret returns information to the program in specific fields. The return fields for installation data are listed and described below.

| Table 11-2. Return Fields |            |            |        |                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------|------------|------------|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Field                     | Dec Offset | Hex Offset | Length | Returned Information                                                                                                                                                                                                                                                                                                                                                                                              |
| TSSRC                     | +76        | +4C        | 2      | Return code. See Table 11-3 for valid codes and their meanings.                                                                                                                                                                                                                                                                                                                                                   |
| TSSSTAT                   | +78        | +4E        | 2      | Status code. See Table 11-4 for valid codes and their meanings.                                                                                                                                                                                                                                                                                                                                                   |
| TSSCRC                    | +80        | +50        | 2      | Character return code. See Table 11-4 for valid codes.                                                                                                                                                                                                                                                                                                                                                            |
| TSSDRC                    | +97        | +61        | 1      | Hex DRC returned from DUFSTR/DUFUPD.                                                                                                                                                                                                                                                                                                                                                                              |
| TSSCSTAT                  | +82        | +52        | 2      | Character status code. See Table 11-5 for valid codes.                                                                                                                                                                                                                                                                                                                                                            |
| TSSRTN                    | +114       | +72        | 256    | <p>When extracting an ACID's installation data, the data is returned by CA-Top Secret in this field.</p> <p><b>Note:</b> When the information returned isn't classified as installation data, TSSRTN is overlaid by the fields listed in Table 11-3.</p> <p>If using a TCPLV4L4 parameter list, then define TSSRTN as 'TSSRTN DS 0CL1024' and the total length of the parameter list should be 'DS XL(1138)'.</p> |

**Non-installation Data Return Fields:** The return fields for non-installation data are listed and described below.

| Table 11-3. Return Fields for Non-Installation Data |            |            |        |                                         |
|-----------------------------------------------------|------------|------------|--------|-----------------------------------------|
| Field                                               | Dec Offset | Hex Offset | Length | Returned Information                    |
| TSSACIDA                                            | +114       | +72        | 8      | ACID name                               |
| TSSFAC                                              | +122       | +7A        | 8      | Facility name                           |
| TSSMODE                                             | +130       | +82        | 8      | Current MODE of user                    |
| TSSTYPE                                             | +138       | +8A        | 8      | ACID type                               |
| TSSTERM                                             | +146       | +92        | 8      | Terminal name                           |
| TSSSYS                                              | +154       | +9A        | 8      | System name (SMF ID)                    |
| TSSACIDF                                            | +162       | +A2        | 32     | Full 32-character ACID name             |
| TSSDEPTA                                            | +194       | +C2        | 8      | Department ACID name                    |
| TSSDEPTF                                            | +202       | +CA        | 32     | Full 32-character name                  |
| TSSDIVA                                             | +234       | +EA        | 8      | Division ACID name                      |
| TSSDIVF                                             | +242       | +F2        | 32     | Full 32-character name of Division ACID |
| TSSZONEA                                            | +274       | +112       | 8      | Zone ACID name                          |
| TSSZONEF                                            | +282       | +11A       | 32     | Full 32-character name of ZONE ACID     |

**Return Codes:** The return codes are listed and described in the following table.

| Table 11-4. Return Codes |          |        |                                                                                                                                                                   |
|--------------------------|----------|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Return Code              | Field    | TSSCRC | Meaning                                                                                                                                                           |
| 00                       | TSSROK   | OK     | Resource is defined/access is granted.                                                                                                                            |
| 04                       | TSSRND   | ND     | Resource isn't defined to CA-Top Secret. If performing a facility check, a return code of 4 means that the facility to be checked isn't defined to CA-Top Secret. |
| 08                       | TSSRNA   | NA     | Resource is defined/access isn't authorized.                                                                                                                      |
| 12                       | TSSRIPL  | IP     | Parameter list is invalid.                                                                                                                                        |
| 16                       | TSSRENV  | EN     | Environment error; CA-Top Secret isn't properly installed within the environment.                                                                                 |
| 20                       | TSSRINAC | IA     | CA-Top Secret isn't active.                                                                                                                                       |
| 24                       | TSSRXS   | XS     | Return data exceeded size of TSSRTN field.                                                                                                                        |
| 28                       | TSSRSEG  | SG     | Incorrect FDT segment name.                                                                                                                                       |
| 32                       | TSSRFDT  | FD     | Incorrect FDT field name.                                                                                                                                         |
| 36                       | TSSRUSR  | US     | Field specified is not a user field.                                                                                                                              |
| 40                       | TSSRGF   | GF     | Storage is not available to complete request.                                                                                                                     |

**Status Return Codes:** The status return codes are listed and described below.

| Return Code | Field   | TSSCSTAT | Meaning                                                  |
|-------------|---------|----------|----------------------------------------------------------|
| 00          | TSSSDEF | DE       | User is defined                                          |
| 04          | TSSSUND | UN       | User is undefined                                        |
| 08          | TSSSNSO | NS       | User isn't signed on                                     |
| 12          | TSSSIDT | ID       | Invalid device type; not a standard IBM 3270 device type |

## 11.1.2 Performing Checks

The CA-Top Secret Application Interface performs various checks based on the information supplied in the Request Record. Each check requires that specific data be supplied.

The Application Interface uses the field TSSHEAD to indicate the parameter format used (for compatibility with programs calling the interface compiled or assembled in previous releases of CA-Top Secret). When compiling or assembling programs that call the Application Interface with the file SYSLIB, the field TSSHEAD should be initialized to the value TCPLV4L4. Other supported values for TSSHEAD are:

| Value        | CA-Top Secret Release |
|--------------|-----------------------|
| TCPLV4L4     | 5.0                   |
| TCPLV4L4     | 4.4                   |
| TCPLV4L3     | 4.3                   |
| TCPLV4L2     | 4.2                   |
| Other values | 4.1                   |

In general, information must be passed for the TSSCLASS, TSSRNAME, and TSSACC fields. If you are issuing DUFUPD, the new information must be supplied in the TSSRTN field.



### 11.1.2.1 General Resource Checking

The interface allows an application program to check resources. To perform a resource check for resources other than data sets or volumes, the following Request Record fields must be supplied by the application program.

| <b>Field</b>    | <b>Content</b>                                                                                                                                                                                                                                                                                          |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>TSSHEAD</b>  | Must be initialized to TCPLV4L4.                                                                                                                                                                                                                                                                        |
| <b>TSSCLASS</b> | A valid resource class name (refer to the RDT for a list of valid resource class names).                                                                                                                                                                                                                |
|                 | <b>Note:</b> Any type of resource can be checked from any environment. For example, an DL/1 PSB can be checked from the CICS environment.                                                                                                                                                               |
| <b>TSSRNAME</b> | The name of the resource to be checked.                                                                                                                                                                                                                                                                 |
| <b>TSSACC</b>   | The requested resource access level.                                                                                                                                                                                                                                                                    |
|                 | Access levels depend on the type of resource being checked. Some resources (such as ABSTRACT, TERMINAL, and APPL), aren't associated with access levels; therefore, TSSACC need not be supplied. Resources and their associated access levels are discussed in Chapter 7, <i>Protecting Resources</i> . |
| <b>TSSCACEE</b> | Must contain a pointer to the user's ACEE if the check is for another ACID.                                                                                                                                                                                                                             |
| <b>TSSPPGM</b>  | If PRIVPGM is being used to protect access to resources, this field must contain the name of the program that must be used to access the resource (the privileged program).                                                                                                                             |
| <b>TSSLOG</b>   | If violation logging is required, code Y. If any other value is entered, no logging will occur.                                                                                                                                                                                                         |

CA-Top Secret will supply a Return Code in the TSSRC field and a Character Return Code in the TSSCRC field to the application program.

### 11.1.2.2 Data Set and Volume Checking

The interface allows an application program to perform security checks for data set names on DASD or TAPE volumes, or to determine user access to DASD or TAPE volumes without regard to specific data set names.

**TSSACC Field Specifications:** The specifications for the TSSACC field are:

1. For DATASET, DASDVOL, and TAPEVOL, use the levels defined by IBM for RACHECK. These levels are: ALTER, CONTROL, UPDATE, and READ.
2. For DATASETX, use these CA-Top Secret extended access levels:

SCRATCH (X'08')  
 CREATE (X'10')  
 WRITE (X'20')  
 READ (X'40')  
 FETCH (X'80')  
 ALL (X'FF')

3. The Parameter list DSECT can be found as #TSSCPL on the Optional Materials File.

**TSSCLASS and RNAME Values:** Depending on the level of access you wish to check, TSSCLASS and RNAME can have the following values.

| TSSCLASS | RNAME  | TSSVOL | TSSACC Choices                           |
|----------|--------|--------|------------------------------------------|
| DATASET  | dsname | volser | ALTER, CONTROL, UPDATE, READ             |
| DASDVOL  | volser | unused |                                          |
| TAPEVOL  | volser | unused |                                          |
| DATASETX | dsname | volser | SCRATCH, CREATE, WRITE, READ, FETCH, ALL |
| DASDVOLX | volser | unused |                                          |
| TAPEVOLX | volser | unused |                                          |
| DATASETT | dsname | volser | unused                                   |
| DASDVOLT | volser | unused |                                          |
| TAPEVOLT | volser | unused |                                          |

### 11.1.2.3 Transaction/Command Checking

The interface allows an application program to perform transaction or panel checking by specifying:

- TSSCLASS name of LCF or TRAN, and
- TSSRNAME containing the transaction or panel name.

The TSSPPGM and TSSACC parameters are **not** required.

CA-Top Secret will supply a Return Code in the TSSRC field and a Character Return Code in the TSSCRC field to the application program.

### 11.1.2.4 Facility Checking

The interface allows an application program to determine if a user is authorized to sign on to a certain facility. For facility checking, the program must supply:

- TSSCLASS name of FACILITY, and
- TSSRNAME containing the name of the facility to be checked.

**Note:** TSSCACEE must contain a pointer to the user's ACEE if the check is being done for another ACID.

This function will provide information on the current user if TSSCACEE isn't coded.

CA-Top Secret will supply the application program with a Return Code in the TSSRC field and a Character Return Code in the TSSCRC field.

**Note:** If the Return Code is equal to 4, the facility name supplied by the application program in the TSSRNAME field isn't defined.

### 11.1.2.5 Facility List Checking

The interface allows an application program to obtain a list of all the facilities that a user is authorized to access. The application program must specify FACLIST in the TSSCLASS Request Record field. No other fields are required.

**Note:** When using FACLIST, you **must** select the TSSPLEN2 length field in the parameter list for the language you are using. The members for each language are listed on 11-3.

CA-Top Secret will supply the application program with a list of all facilities that the user is authorized to access. Each facility name is eight characters long, padded with blanks. The facility list is returned in the TSSRTN field of the Return Record. The information pertains only to the signed-on user.

**Note:** If using a TCPLV4L4 format parameter list, then define TSSRTN as 'TSSRTN DS 0CL1024' and the total length of the parameter list should be 'DS XL(1138)'.

### 11.1.2.6 Resource List Checking

The RESLIST function of TSSAI is used to generate a list of resources that are added or permitted to the requesting user within a specified resource class. The RESLIST function for a resource can be protected because RESLIST is defined by CA-Top Secret to the RDT. The TSS ADD command function

```
TSS ADD(owner) RESLIST(resource)
```

establishes protection for the RESLIST function on the named *resource*.

Alternatively, all RESLIST **resource** access can be protected by adding the DEFPROT attribute to the RESLIST *resource*.

RESLIST access can be permitted or denied in the following way:

```
TSS PERMIT(user) RESLIST(resource)
TSS PERMIT(user) RESLIST(resource) ACTION(DENY,FAIL)
```

To invoke the RESLIST function through the application interface, the field TSSCLASS of the parameter TSSCPL must be set to the value RESLIST, and the name of the *resource* must be placed in the field TSSRNAME. Normally, the request is performed for the ACID assigned to the calling program. However, the user may provide an ACEE pointer for a specified user by moving the pointer value to TSSCACEE prior to the TSSAI call.

The length for TSSRTN returned by RESLIST is 1024; correspondingly, the length of TSSCPL is stored in TSSPLEN2, similar to the length used by the FACLIST function.

TSSRTN is formatted in the TSSCPLA Assembler copybook in the TSSOPMAT installation file. RESLIST formats the TSSRTN structure into the following:

**TSSRFLDS** a four-byte count of the number of entries returned

**TSSRDATA** a list structure

The entries of the TSSRDATA list structure are mapped as follows:

**TSSRLLEN** a two-byte resource name length

**TSSRLGEN** a one-byte indicator showing

**TSSRLRES** up to 255 character resource name whose actual length is contained in TSSRLLEN

The TSSRLENT Assembler DSECT has been provided in the TSSCPLA copybook for programming convenience in mapping these entries.

TSSRTN has been formatted for the COBOL programmer in TSSCPLC, and for the PL/I programmer in TSSCPLP. Please refer to the comments in the copybooks for programming suggestions in higher level languages.

TSSRC can return all of the values previously documented in this chapter under the label *Return Codes*. However, the RESLIST call can also return the special value 24 (TSSCRC='XS') which means the return area is too small to hold list.

### 11.1.2.7 Logging User Information

The interface allows an application program to log up to 40 bytes of user-supplied data. The data is logged to the Audit/Tracking File and/or SMF based on the installation-defined logging options.

To log information, the application program must supply the following information to the interface:

- Specify a TSSCLASS name of LOG.
- The information to be logged must be supplied in the first 40 bytes of the TSSRTN request field.

**Note:** TSSCACEE must contain a pointer to the user's ACEE if the check is being done for another ACID.

### 11.1.2.8 Dynamic Update Facility

The Application Interface can be used to extract and/or update user-dependent data such as counts, balances, totals, CPU usage, and so on. CA-Top Secret stores this data in the installation data area, as created by the TSS CREATE, TSS ADD, or TSS REPLACE command functions. To use the TSSAI functions DUFUPD and DUFXTR, an ACID must have the TSS attributes, DUFUPD and DUFXTR, respectively.

The installation data, INSTDATA, is maintained in the user's Security Record on the Security File. The installation data can be updated or extracted for any ACID on the Security File. However, TSSAI can't be used to REMOVE an ACID's INSTDATA by placing spaces or hex-zeros in the TSSRTN field. You must use the TSS REMOVE command to eliminate INSTDATA.

Although INSTDATA may be added to a profile record, its use is limited. INSTDATA on a profile is only available when using DUFXTR for the current user. When specifying DUFUPD or DUFXTR for another user, the INSTDATA must be on the user's record, not in a profile.

**Extracting Installation Data:** To extract the installation data for the active user, the application program must supply the following information to the interface:

- Specify a TSSCLASS name of DUFXTR.
- Do **not** specify anything for TSSRNAME; this field must contain blanks.

To extract the installation data for another user, the application program must supply the following information to the interface:

- Specify a TSSCLASS name of DUFXTR.
- Specify the ACID name in the TSSRNAME field, left justified.
- For the current user to be authorized to DUFXTR another user, he must be permitted to submit that ACID as follows:

**TSS PER(dufuser) ACID(tssrname)**

| TSSRC | TSSCRC | TSSRTN   | Comments                                                                |
|-------|--------|----------|-------------------------------------------------------------------------|
| 0     | OK     | INSTDATA | 256 characters                                                          |
| 4     | ND     | SPACES   | Active user is undefined, or is authorized but no INSTDATA is available |
| 8     | NA     | SPACES   | Active user not authorized                                              |

CA-Top Secret will return the installation data to the application program in the TSSRTN field.

**Note:** CA-Top Secret returns the DRC for an DUFUPD/DUFXTR violations in field TSSDRC at offset X'61' of TSSCPL.

**Updating the Installation Data:** An ACID's installation data can be updated using DUFUPD. DUFUPD replaces the ACID's existing data. It doesn't change the length of the data, and can't be used to add installation data to an ACID that has none.

**Note:** The length of the installation data can only be changed using the TSS command interface.

To update the installation data for the signed-on user, the application program must supply the following information to the interface:

- Specify a TSSCLASS name of DUFUPD.
- Leave the TSSRNAME field blank unless the Security File is to be updated immediately. When performing an immediate update, enter the appropriate information in this field. Leaving the field blank in this situation causes an in-memory call to occur and INSTDATA won't be updated in the Security File until the user signs off.
- Supply the updated installation data in the TSSRTN field.

To update the installation data for another user, the application program must supply the following information to the interface:

- Specify a TSSCLASS name of DUFUPD.
- Specify the ACID name in the TSSRNAME field.

- For the current user to be authorized to DUUPD another user, he must be permitted to submit that ACID as follows:

**TSS PER(dufuser) ACID(tssrname)**

- Place the updated installation data in the TSSRTN field.

CA-Top Secret will update the installation data by replacing the old data with the supplied data.

### 11.1.2.9 Field Extract and Update Facility

The Application Interface can be used to extract and/or update fields attached to an ACID. All user-defined fields can be extracted and/or updated, as well as many of the fields in the FDT.

**Note:** Predefined fields (like TSO and CICS) are not eligible for TSSAI processing.

To extract field data, the User ACID must have the DUFXTR attribute; the DUFUPD attribute is required to update field data.

**Extracting Field Data:** To extract the data for a specific field:

- Specify a TSSCLASS name of FLDXTR.
- Specify the segment name in TSSPPGM.
- Specify the field name in the TSSRNAME field.

CA-Top Secret will return a word containing the length of the field followed by the field data in the TSSRTN field.

#### Caution Warning

The TSSRTN field should be large enough to contain the returned data. CA-Top Secret Release 3.0 and above supports up to 1024 bytes of data for the field. Some user-defined fields may exceed this length.

**Updating the Field Data:** To update the data of a specific field:

- Specify a TSSCLASS name of FLDUPD.
- Specify the segment name in TSSPPGM.
- Specify the field name in the TSSRNAME field.
- Specify a four-byte word to contain the length of the field, and then specify the data to be stored in the TSSRTN field. (The four-byte length goes in front of the data.)

CA-Top Secret will update the field data by replacing the old data with the supplied data.

### 11.1.2.10 Retrieving an ACID

The interface allows an application program to retrieve the eight-character ACID of the signed-on user, as well as other information. Use the following information to retrieve the indicated data.

- The application program must supply a TSSCLASS name of ACIDNAME.
- No other fields are required.

The Application Interface will return the following data:

| <b>Return Field</b> | <b>Data Description</b>                                                                                                  |
|---------------------|--------------------------------------------------------------------------------------------------------------------------|
| <b>TSSACIDA</b>     | The eight-character ACID of the current user.                                                                            |
| <b>TSSMODE</b>      | The eight-character MODE name of the signed on ACID.                                                                     |
| <b>TSSFAC</b>       | The eight-character name of the facility that the user is currently signed on to.                                        |
| <b>TSSTYPE</b>      | The type of user ACID (USER, DCA, VCA, or SCA).                                                                          |
| <b>TSSTERM</b>      | The name of the terminal the user is signed on to.                                                                       |
| <b>TSSSYS</b>       | The System ID (SMF ID) of the system the user is signed on to. The SMF ID returns left justified and padded with blanks. |

### 11.1.2.11 Other Checks for the ACID

The Application Interface can also perform various other checks for the signed-on user by specifying special keywords in the TSSCLASS field. The checks are listed in the following table.

**Note:** TSSCACEE must contain a pointer to the user's ACEE if the check is being done for another ACID.



| Table 11-6. Other Checks |              |                                                                                                                                                                                                                                                                                                                                                                                                       |
|--------------------------|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| TSSCLASS                 | Return Field | Returned Data Description                                                                                                                                                                                                                                                                                                                                                                             |
| ACIDFULL                 | TSSACIDF     | Full 32 character ACID's name                                                                                                                                                                                                                                                                                                                                                                         |
| DEPTNAME                 | TSSDEPTA     | 8 character name of the ACID's department                                                                                                                                                                                                                                                                                                                                                             |
| DEPTFULL                 | TSSDEPTF     | Full 32 character ACID's department name                                                                                                                                                                                                                                                                                                                                                              |
| DIVNAME                  | TSSDIVA      | 8 character name of the ACID's division                                                                                                                                                                                                                                                                                                                                                               |
| DIVFULL                  | TSSDIVF      | Full 32 character ACID's division name                                                                                                                                                                                                                                                                                                                                                                |
| ZONENAME                 | TSSZONEA     | 8 character name of the ACID's zone                                                                                                                                                                                                                                                                                                                                                                   |
| ZONEFULL                 | TSSZONEF     | Full 32 character ACID's zone name                                                                                                                                                                                                                                                                                                                                                                    |
| ACIDCHK                  | TSSCRC       | <p>Determines whether the ACID in TSSRNAME can be used in the JOB card of a batch job submitted from the current facility, mode, and user. If the submission is permitted, TSSRC=0 and TSSCRC='OK'. If submission is rejected, TSSRC=8 and TSSCRC='NA'.</p> <p>The programmer can use both TSSRC and TSSCRC to determine success; however, only TSSCRC is listed as a return field in this table.</p> |

**Note:** If the header of the parameter list indicates a TCPLV4L2 or previous format, the ACIDFULL, DEPTFULL, and DIVFULL fields will be truncated to the first 20 characters.

### 11.1.3 Programming Examples

There are three languages your application program can be written in; Assembler, PL/1, or COBOL. This section describes the technique that should be used to invoke the Application Interface for each language.

**Note:** These examples apply to the BATCH facility.

#### 11.1.3.1 Assembler

In Assembler, use the LOAD MACRO to load TSSAI. Then, BRANCH and LINK to the routine. At the close of processing, issue the DELETE MACRO to delete the module.

**Note:** An alternate approach when using Assembler is to use the LINK MACRO.

File TSSOPMAT on the distribution tape contains examples of Assembler code that illustrate both techniques for BATCH, STC, and TSO facilities; member EXAIBA1 shows the use of LOAD; member EXAIBA2 shows the use of LINK.

```

 USING TSSCPL,R2
 LA R2,TSSCPL
 VIOLATION LOGGING REQUESTED
 MVC TSSHEAD,=CL8'TCPLV4L4'
 MVC TSSCLASS,=CL8'DUFSTR' REQUEST INSTALLATION DATA
 LA R1,TSSCPL R1 -> PARAMETER LIST
 ST R1,ATSSCPL
 LA R1,ATSSCPL
 LINK EP=TSSAI LINK TO APPLICATION INTERFACE
 LTR R15,R15 Q RETURN CODE OK
 BNZ NOTOK A NOT OK GO DO YOUR ERROR
OK DS OH DO WHAT YOU LIKE
 .
 .
 .
ATSSCPL DS A ADDRESS OF PARAMETER LIST
 COPY TSSCPLA
 #TSSCPL,TYPE=CSECT MACRO MAPS APPLICATION
 . INTERFACE PARM LIST
 .

```

*Example 11-1. Assembler***11.1.3.2 PL/I**

PL/I programs should use a dynamic CALL function to invoke the Application Interface. File TSSOPMAT on the distribution tape contains sample code for PL/I in the member named EXAICP1.

```

PLIEX1: PROC OPTIONS (MAIN);
 DCL TSSAI EXTERNAL ENTRY OPTIONS (ASM INTER);
 %INCLUDE (TSSCPLP); /* INCLUDE DEFINITIONS OF PLIST */
 .
 .
 TSSHEAD='TCPLV4L4';
 TSSLOG='N'; /* DON'T LOG VIOLATIONS */
 TSSCLASS='DUFSTR'; /* REQUEST INFORMATION DATA */
 CALL TSSAI (TSSCPL);
 .
 .
 .

```

*Example 11-2. PL/I*

### 11.1.3.3 COBOL

COBOL programs should use a dynamic `CALL` function to invoke the Application Interface, and a `CANCEL` function to delete the requested module. File `TSSOPMAT` on the distribution tape contains COBOL coding examples that use a dynamic call to invoke the Application Interface in the member named `TSSAIIC1`.

```

000010 IDENTIFICATION DIVISION.
.
.
001000 DATA DIVISION.
.
.
001010*****
001020*
001030* ENSURE TSSCPL IS IN THE COBOL COPY LIBRARY OR COPY YOUR OWN *
001040* VERSION INLINE INTO PROGRAM, USE OF COPY LIBRARY IS *
001050* RECOMMENDED SO THAT CHANGES WILL ONLY REQUIRE RECOMPILE *
001060* AND NOT RECODING OF SOURCE *
001070*****
001080 01 TSSCPL COPY TSSCPLC.
001090 77 TSSAI PICTURE X(8).
.
.
002000 PROCEDURE DIVISION.
.
.
010010*****
010020*
010030* CALL CA-Top Secret APPLICATION INTERFACE TO EXTRACT *
010040* INSTALLATION DATA FROM USER'S SECURITY RECORD *
010050* VIOLATION LOGGING IS REQUESTED *
010060*****
010070 MOVE 'TCPLV4L4' to TSSHEAD
010080 MOVE 'Y' to TSSLOG
010090 MOVE 'DUFXTX' to TSSCLASS.
010100 MOVE 'TSSAI' to TSSAI.
010110 CALL TSSAI USING TSSCPLC.
010120 IF NOT TSSROK GO TO 100-CHECK-TOP-RETURN
010130 MOVE TSSRTN TO USER-AREA-FOR-INST-DATA.
.
.
020010*****
020020*
020030* CANCEL CA-Top Secret APPLICATION INTERFACE DO THIS WHEN THE *
020040* CA-Top Secret INTERFACE IS NO LONGER REQUIRED BY PROGRAM AND *
020050* THUS FREE UP THE STORAGE IT OCCUPIES IF NOT IN COMMON STORE *
020060*
020070*****
020090 MOVE 'TSSAI' to TSSAI.
020100 CANCEL TSSAI.
.
.

```

*Example 11-3. COBOL*

## 11.2 Extending Security Through the VSE Security Interface

The VSE Standard Security Interface in VSE/ESA 2.4 and above, is used to drive access authorization checking for CA-Top Secret. Because CA-Top Secret supports the Standard Security Interface as a means of implementing security, it is completely compatible with, and transparent to, IBM software that requires security checking. The VSE Standard Security Interface, therefore, can be used to perform specialized security checking with CA-Top Secret.

Although the macros that comprise the original VSE Standard Security Interface are fully supported by CA-Top Secret Release 3.0 and above, it is recommended that all security checking be performed using the VSE SAF interface. The macro that can be used to interface with CA-Top Secret is:

**RACROUTE** The centralized SAF macro for all security calls.

The examples given in this section reference only the VSE SAF interface macros, but equivalent functionality can be obtained by using the VSE Standard Security Interface versions. You can use the standard IBM security macros to:

- validate access to all standard resources recognized by CA-Top Secret.
- validate access to an installation-defined resource (for example, CPUs).
- provide security support for your tape management system.

The following Standard Security Interface (SU32) forms are supported. However, it is strongly recommended that the RACROUTE macro be used in their place. The macros and their associated RACROUTE forms are shown below.

**For This Macro Use This RACROUTE Form**

**FRACHECK** RACROUTE REQUEST=FASTAUTH

**RACDEF** RACROUTE REQUEST=DEFINE

**RACHECK** RACROUTE REQUEST=AUTH

**RACINIT** RACROUTE REQUEST=VERIFY

**RACXTRT** RACROUTE REQUEST=EXTRACT

**Note:** The security macro, RACLIST, although used by CICS and DL/1 for security initiation, isn't available for user customization.

## 11.2.1 The RACROUTE Macro Calls

RACROUTE is part of the System Authorization Facility (SAF) of the VSE Security Interface; it is available for all releases of VSE currently supported by CA-Top Secret. RACROUTE invokes the VSE/SAF router (provided by CA-Top Secret), which in turn invokes the appropriate service routine(s).

CA-Top Secret supports the following RACROUTE calls:

|                         |                                                               |
|-------------------------|---------------------------------------------------------------|
| <b>REQUEST=AUTH</b>     | Verify resource access                                        |
| <b>REQUEST=DEFINE</b>   | Not normally available for customization                      |
| <b>REQUEST=EXTRACT</b>  | Obtain user information and perform encryption services       |
| <b>REQUEST=FASTAUTH</b> | Verify resource access through a fast path mechanism          |
| <b>REQUEST=LIST</b>     | Not normally available for customization                      |
| <b>REQUEST=VERIFY</b>   | Validate userid and password, and create security environment |

### 11.2.1.1 The RACROUTE REQUEST=VERIFY Call

RACROUTE REQUEST=VERIFY requests authorization checking for a job/session/STC initiation, and can be used to support signon validation for individual users, including password validation. In addition, RACROUTE REQUEST=VERIFY can be used to create a temporary security environment for third party authorization checking.

In response to a RACROUTE REQUEST=VERIFY call, CA-Top Secret builds an Accessor Environment Element (ACEE) and loads security record(s) if the initiation is allowed. These records are then used for further security checks during the job/session.

The following example shows how to define the RACROUTE REQUEST=VERIFY call to validate user signon.

|                          |                                      |
|--------------------------|--------------------------------------|
| RACROUTE REQUEST=VERIFY, | (Specify RACROUTE request type)      |
| ENVIR=CREATE,            | (Request Signon/Environment Create)  |
| USERID=uid-loc,          | (Identifies user)                    |
| PASSWRD=pw-loc,          | (User's password, if supplied)       |
| NEWPASS=newpw-loc,       | (User's new password, if supplied)   |
| TERMINID=term-loc,       | (Terminal name)                      |
| ACEE=acee-anchor,        | (See discussion in next section)     |
| WORKA=work-addr          | (512 byte work area required by SAF) |

*Example 11-1. Validate Signon*

RACROUTE REQUEST=VERIFY is also called at job/session/STC termination to delete the security environment ACEE. The following example shows how to define the RACROUTE REQUEST=VERIFY call to process user logoff.

|                          |                                      |
|--------------------------|--------------------------------------|
| RACROUTE REQUEST=VERIFY, | (Specify RACROUTE request type)      |
| ENVIR=DELETE,            | (Request deletion of ACEE / signoff) |
| ACEE=acee-anchor         | (Provide ACEE anchor address)        |

*Example 11-2. Process User Logoff*

**ACEE= Parameter Requirements:** For the RACROUTE REQUEST=VERIFY call, the following requirements apply in coding or omitting the ACEE= parameter:

1. If you want CA-Top Secret to manage the ACEE, then omit the ACEE= keyword. CA-Top Secret will anchor the ACEE. In this case, it will always reside below the 16 megabyte line (24 bit storage).
2. If you don't want CA-Top Secret to manage (and, therefore, automatically locate) the ACEE, then code ACEE=. The address you are providing is a location (that is, a place holder) that CA-Top Secret will use to return the address of the ACEE for a successful RACROUTE REQUEST=VERIFY with ENVIR=CREATE. Similarly, in the case of a RACROUTE REQUEST=VERIFY with ENVIR=DELETE, the ACEE= parameter is the address of the anchor that points to the actual ACEE to be deleted. For all other RACROUTE requests, the ACEE parameter points to the ACEE itself.

The ACEE will reside above the 16 megabyte line only when ACEE= has been coded and the caller is executing in a 31 bit addressing mode.

### 11.2.1.2 The RACROUTE REQUEST=AUTH Call

RACROUTE REQUEST=AUTH requests authorization for use of a specified resource at a specific access level. It is primarily used for data set and volume authorizations, although it can be used for any resource class that would be available using RACROUTE REQUEST=FASTAUTH.

The following example shows how to define the RACROUTE REQUEST=AUTH call to validate data set access.

```

RACROUTE REQUEST=AUTH,
 CLASS='DATASET',
 ENTITY=DSNAME,
 ATTR=attr-name, (read/update/alter)
 WORKA=WORKAREA

DSNAME DC CL44 'SAMPLE.DATA.SET.NAME '
WORKAREA DS XL512

```

*Example 11-3. Validate Data Set Access*

The following example shows how to define the RACROUTE REQUEST=AUTH call to log data to the Audit/Tracking File.

```

RACROUTE REQUEST=AUTH,
 CLASS=CLASSNML,
 ENTITY=EVENT3, (log data area)
 WORKA=RWORK

CLASSNML DC AL1 (L 'CLASSNM)
CLASSNM DC C 'USERLOG '
EVENT3 DC AL1 (21),CL21 'EVENT 3 HAS COMPLETED '
RWORK DS XL512

```

*Example 11-4. Log Data to the Audit/Tracking File*

Resources are classified by Class Names. The following table lists valid Class Names, their functions, and any required data.

| Table 11-7 (Page 1 of 2). RACROUTE REQUEST=AUTH Class Names |                                                                                         |                                |
|-------------------------------------------------------------|-----------------------------------------------------------------------------------------|--------------------------------|
| Class Name                                                  | Function                                                                                | Entity                         |
| ABSTRACT                                                    | Validate access to user resource                                                        | 8 character user resource name |
| CHGPROP                                                     | Cross-CPU change propagation                                                            | Recovery record buffer         |
| DASDVOL                                                     | Validate access to DASD volumes                                                         | 1-6 character volume serial    |
| DASDVOLT                                                    | Determine if DASD volume access should be granted in any manner; RC=8 if not accessible | 1-6 character volume serial    |
| DATASET                                                     | Validate access to a data set                                                           | 44 character dsname            |



| Table 11-7 (Page 2 of 2). RACROUTE REQUEST=AUTH Class Names |                                                                                                                                                                                                                        |                                                        |
|-------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------|
| Class Name                                                  | Function                                                                                                                                                                                                               | Entity                                                 |
| DATASET                                                     | Determine if data set can be used with any access level. RC=0 is returned if any permission is found; RC=8 is returned if no permission is found.                                                                      | 44 character dsname                                    |
| DATSETXa                                                    | Extended data set check<br>CLASSNAME+8 contains a one byte access mask, "a", that overrides the ATTR values:<br><br>x'08' = SCRATCH<br>x'10' = CREATE<br>x'20' = WRITE<br>x'40' = READ<br>x'80' = FETCH<br>x'FF' = ALL | 44 character dsname                                    |
| DUFXTR                                                      | Obtain INSTDATA for an ACID                                                                                                                                                                                            | See Dynamic Extract/Update Facility section            |
| DUFUPD                                                      | Replace INSTDATA date                                                                                                                                                                                                  | See Dynamic Extract/Update Facility section            |
| INSEXIT                                                     | Invoke installation exit<br>Site call                                                                                                                                                                                  | Site-dependent data                                    |
| LOCK                                                        | Unconditionally lock a terminal to prevent access                                                                                                                                                                      | none                                                   |
| TAPEVOL                                                     | Validate access to tape volume                                                                                                                                                                                         | 1-6 character volume serial                            |
| TAPEVOLT                                                    | Determine if tape volume can be used in any manner; RC=8 if not accessible                                                                                                                                             | 1-6 character volume serial                            |
| UNLOCK                                                      | Conditionally unlock a terminal                                                                                                                                                                                        | Password                                               |
| USERLOG                                                     | Log site-dependent data                                                                                                                                                                                                | 1-byte length field followed by up to 44 bytes of data |
| Any resource defined in the RDT                             | Installation defined resources                                                                                                                                                                                         | Length is defined in the RDT entry                     |

### 11.2.1.3 Dynamic Extract/Update Facility

The Dynamic Extract/Update Facility (DUF) allows the installation to extract and update user-dependent data such as counts, statistics, totals, CPU usage, and so on. This data is the INSTALLATION data area as created by the TSS CREATE or TSS ADD INSTDATA command function.

**Note:** The INSTDATA area can't be extended through DUF. Extension of the INSTDATA field can be done only through a TSS REPLACE or TSS ADD INSTDATA command function.

The installation data is maintained in the user's Security Record on the Security File. Installation data for any ACID can be maintained using DUF:

- the current ACID as defined by an ACEE, or
- any ACID on the Security File.

The RACROUTE REQUEST=AUTH call can be used to extract, maintain, and update the installation data (INSTDATA). For this function, either the program name must be defined by the DUFPGM control option or the caller of these functions must be authorized through the TSS ADD command function parameters DUFXTR and/or DUFUPD.

**Note:** To display INSTDATA, it must be maintained in EBCDIC format and not in binary or hexadecimal values.

An example of meaningful INSTDATA is shown below.

```
INSTDATA(' ID=Q477,CPU=00104, CODE=A, BAL=$00366.00')
```

The Class Names and their functions are listed in the following table.

| Class Name | Function                                                                                                                                                                                                                   | Entity                                               |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------|
| DUFXTR     | Extract the INSTDATA for an ACID.<br><br>If the ACID name is left blank, the INSTDATA of the <b>active</b> user is extracted.<br><br>If the ACID name is supplied, the INSTDATA for the <b>supplied</b> ACID is extracted. | +0(8) ACID name or blanks<br>+8(255) INSTDATA buffer |

| Class Name | Function                                                                                                                                                                                                                                                                                   | Entity                                                          |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------|
| DUFUPD     | <p>Replaces an ACID's INSTDATA.</p> <p>If the ACID name is left blank, the INSTDATA of the <b>active</b> user is replaced.</p> <p>If the ACID name is supplied, the INSTDATA for the <b>supplied</b> ACID is replaced.</p> <p>DUF updates will be propagated to all CPF-defined nodes.</p> | <p>+0(8) ACID name or blanks</p> <p>+8(255) INSTDATA buffer</p> |

The ACID field of the entity (offset 0) indicates whether an in-memory or an on-file extract/update will take place. If the ACID field is blank, then an in-memory extract/update occurs based on the executing user's ACID and security record. Otherwise, the INSTDATA for the ACID name is obtained from the Security File and used.

In-memory INSTDATA is chained from the ACEE for the current ACID in RACF-compatible format. Its format is:

|                                                                                                         |
|---------------------------------------------------------------------------------------------------------|
| <p>+0(1) length of INSTDATA including this length byte</p> <p>+1(?) installation data for this user</p> |
|---------------------------------------------------------------------------------------------------------|

**Note:** When used with DUF, the length must not appear in the entity field reserved for the installation data.

CA-Top Secret automatically updates INSTDATA upon job/session termination if it was previously changed in-memory. The following example shows an in-memory extract/update.

```

* Update some code
* Note that this is an in-memory update
* Installation data will be automatically updated on file when the
* current session ends
MVC INSTDATA+0(8),=8C' ' indicate in-memory extract/update
MVC RAUHL,RAUHSKL transfer skeleton
RACROUTE REQUEST=AUTH,
 CLASS='DUFXR',
 ENTITY=INSTDATA,
 WORKA=TSSWORK,
 MF=(E,RAUHL)
LTR 15,15 successful extract?
BNZ DUFERROR branch if unsuccessful
MVI INSTDATA+23,C'A' set operation code
RACROUTE REQUEST=AUTH,
 ENTITY=INSTDATA,
 MF=(E,RAUHL)

```

The installation data area contains the ACID followed by data:

```

INSTDATA DS CL8,CL255 acid and data
RAUHL RACROUTE REQUEST=AUTH, Skeletal parameter list
 CLASS='DUFUPD',
 MF=L

```

To perform an INSTDATA extract/update for any ACID on the Security File, the first line of the above code would read:

```

MVC INSTDATA+0(8),=CL8'acid' acid is the name of a
 particular ACID

```

**Error Return Codes:** If feedback is used, possible detail error return codes that may be produced are:

- x'1E' Invalid parameter
- x'1F' No DUFXTR/DUFUPD authority
- x'46' ACID undefined
- x'4E' ACID has no INSTDATA

#### 11.2.1.4 Tips for Using DUF

- Don't try to reduce or extend the size of the INSTDATA. The only way to change the size of INSTDATA is by using the TSS REPLACE or TSS ADD INSTDATA command function.
- To enable the TSS command to display meaningful information, maintain EBCDIC information and not binary data.
- Keep counters in unpacked numeric format.
- Use fixed format fields or user free-format with identifiers.
- Use your installation exit (TSS command call) to enforce INSTDATA standards. When updating, the entire INSTDATA field must be replaced.

#### 11.2.1.5 The RACROUTE REQUEST=FASTAUTH Call

RACROUTE REQUEST=FASTAUTH can be used to validate access to resources, user resources, fields, and LCF commands. RACROUTE REQUEST=FASTAUTH is a fast path mechanism that should be used by online facilities to minimize performance impact. Logging of violations and activity to SMF or the Audit/Tracking File is automatically provided by the RACROUTE REQUEST=FASTAUTH interface through indirect CSA buffering with the CA-Top Secret address space.

##### Caution

If the FRACHECK macro is used instead of RACROUTE REQUEST=FASTAUTH, the contents of general purpose registers 0 through 5, 14 and 15 are destroyed and not restored by FRACHECK.

### 11.2.1.6 Class Name Formats

CA-Top Secret accepts the class name for a RACROUTE REQUEST=FASTAUTH in two different formats. They are:

1. as a character string—eight characters with blank padding. For example:

ABSTRACT, PROGRAM, TERMINAL

2. as a hexadecimal value string that ends with a question mark (?). When specified in this format, the fifth byte must contain the hexadecimal resource code for the desired class as defined in the RDT. Examples are:

ABS-UUU?, PGM-PPP?, TER-TTT?

You can code the class name in either format. However, the format used also effects how the access level and privilege program (PRIVPGM) information is specified.

### 11.2.1.7 Specifying a Character String Class Name

When the class name is specified as a character string, access level information is passed to RACROUTE REQUEST=FASTAUTH through the ATTR= keyword. The RACROUTE macro supports the following four values for ATTR=:

|                |                                                                                 |
|----------------|---------------------------------------------------------------------------------|
| <b>ALTER</b>   | Requests full (ALL) access                                                      |
| <b>CONTROL</b> | Requests control level access                                                   |
| <b>UPDATE</b>  | Requests update level access                                                    |
| <b>READ</b>    | Requests read level access, and is the default when no value has been specified |

If these values are passed by a register, then the value in the specified register must match the ATTR= values as documented in the table below. In processing the ATTR= values, the value coded is translated to the following CA-Top Secret internal access level value. To be properly interpreted, when you define your own resource classes, you should ensure that the access levels defined match the following values for ATTR=.

| ATTR= Keyword | ATTR= Hex value | Access Level | Access Level Bit Value |
|---------------|-----------------|--------------|------------------------|
| ALTER         | X'00000080'     | X'FFFF'      | B'11111111,11111111'   |
| CONTROL       | X'00000008'     | X'0400'      | B'00000100,00000000'   |
| UPDATE        | X'00000004'     | X'8000'      | B'10000000,00000000'   |
| READ          | X'00000002'     | X'4000'      | B'01000000,00000000'   |

The following table lists valid Class Names, their functions, and the required data. Class Names of general owned resources, listed in Table 11-2, are identified by a ? in position 8 (offset +7).

| Table 11-9 (Page 1 of 2). RACROUTE REQUEST=FASTAUTH Class Names |                                                                                                                                                                            |                                                                           |
|-----------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------|
| Class Name                                                      | Function                                                                                                                                                                   | Entity                                                                    |
| ABSTRACT                                                        | Validate access to ABSTRACT user resources                                                                                                                                 | 8 character abstract resource name                                        |
| XLCFCMD<br>XLCFXCTN<br>LCF                                      | Determine if the command, transaction, monitor, or panel is owned as an OTRAN. If it is, perform an OTRAN check; if it isn't, access to the resource as an LCF is checked. | 8 character resource name                                                 |
| ABS-UUU?                                                        | Validate access to an ABSTRACT resource *                                                                                                                                  | +0(8) resource name<br>+8(1) access mask<br>+9(8) privileged program name |
| APL-AAA?                                                        | Validate access to an DL/1 application                                                                                                                                     | Same as above                                                             |
| AREAbbb?                                                        | Validate access to a CA-IDMS database area                                                                                                                                 | Same as above                                                             |
| CP-888?                                                         | Validate access to VM CP commands                                                                                                                                          | Same as above                                                             |
| DBD-ddd?                                                        | Validate access to DL/1 DBD                                                                                                                                                | Same as above                                                             |
| DCT-EEE?                                                        | Validate CICS destination table                                                                                                                                            | Same as above                                                             |
| DIAG999?                                                        | Validate VM diagnose codes                                                                                                                                                 | Same as above                                                             |
| FCT-FFF?                                                        | Validate CICS FCT                                                                                                                                                          | Same as above                                                             |
| FLD-RRR?                                                        | Validate database field level                                                                                                                                              | Same as above                                                             |
| GUR-MMM?                                                        | General use; UR1                                                                                                                                                           | Same as above                                                             |
| GUR-NNN?                                                        | General use; UR2                                                                                                                                                           | Same as above                                                             |
| JCT-JJJ?                                                        | Validate journal control table                                                                                                                                             | Same as above                                                             |

| Table 11-9 (Page 2 of 2). RACROUTE REQUEST=FASTAUTH Class Names |                                                                                                                                                                                                                                                                                   |                           |
|-----------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------|
| Class Name                                                      | Function                                                                                                                                                                                                                                                                          | Entity                    |
| LCF                                                             | Check for OTRAN ownership of the command, transaction, monitor, or panel. If owned, perform an OTRAN resource check to determine if the user has access to the OTRAN resource. If unowned, perform an LCF resource check to determine if the user has access to the LCF resource. | 8 character resource name |
| LCFONLY                                                         | Determine if the user has access to the command, transaction, monitor, or panel as an LCF resource. An OTRAN resource check isn't performed here.                                                                                                                                 | Same as above             |
| OTRAN                                                           | Check for OTRAN ownership of the command, transaction, monitor, or panel. If owned, perform an OTRAN resource check to determine if the user has access to the OTRAN resource.                                                                                                    | Same as above             |
| NET-000?                                                        | Validate VM RSCS nodename                                                                                                                                                                                                                                                         | Same as above             |
| PGM-PPP?                                                        | Validate O/S programs                                                                                                                                                                                                                                                             | Same as above             |
| PPT-QQQ?                                                        | Validate CICS transactions                                                                                                                                                                                                                                                        | Same as above             |
| PSB-SSS?                                                        | Validate DL/I PSG                                                                                                                                                                                                                                                                 | Same as above             |
| SUB-aaa?                                                        | Validate CA-IDMS subschema                                                                                                                                                                                                                                                        | Same as above             |
| TRM-TTT?                                                        | Validate network terminal ID                                                                                                                                                                                                                                                      | Same as above             |
| TST-ZZZ?                                                        | Validate CICS temporary storage table                                                                                                                                                                                                                                             | Same as above             |
| USERxx                                                          | Validate unowned user resource                                                                                                                                                                                                                                                    | +0(8) resource name       |

\* For additional information about the ABSTRACT resource access mask refer to TSS.OPTIONAL.MATERIAL(TSSINST1) "Resource Class Mapping" on the distribution tape.

The following table lists the required data for user-defined resources. These resources are created by adding them to the Resource Descriptor Table (RDT), and specifying the particular resource class name.



| Table 11-10. RACROUTE REQUEST=FASTAUTH For User-Defined Class Names |                                                                                                         |
|---------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------|
| Class Name                                                          | Entity                                                                                                  |
| User-Defined Resource in the RDT Record                             | +0(8) or +0(44) resource name<br>+8(1) or +44(1) access mask<br>+9(8) or +45(8) privileged program name |

To support 44 character lengths, the security administrator must also attach the LONG attribute. For example, if a security administrator wants to create a resource class name called @RESOURZ and wants it to be 44 characters in length, enter:

```
TSS ADD(RDT) RESCLASS(@RESOURZ) RESCODE(hex code)
ATTR(LONG)
```

### 11.2.1.8 Performance Shortcuts

With this type of Class Name, CA-Top Secret typically first determines if the resource is protected by checking for ownership. If the resource isn't owned/protected, RACROUTE REQUEST=FASTAUTH returns a Return Code of 04 (accessible but not protected).

To increase system performance, a faster path can be taken. CA-Top Secret assumes that a resource is owned/protected, if a ? is placed in position 5 (offset +4) of the Class Name. For example:

- If GUR-NNN? is specified as the Class Name, CA-Top Secret checks for ownership of the resource.
- If GUR-?NN? is specified as the Class Name, CA-Top Secret assumes that the resource is protected; no ownership checking is performed. In effect, the specification of a question mark (?) forces the DEFPROT attribute to be used for the resource.

### 11.2.1.9 Return Codes

| Return Code | Meaning                           |
|-------------|-----------------------------------|
| 00          | Access allowed, resource defined. |
| 04          | Resource not defined.             |
| 08          | Access denied.                    |

### 11.2.1.10 Sample RACROUTE REQUEST=FASTAUTH Specification

This example is written for a multiple user address space. The ACEE= parameter isn't needed for a single user address space.

```

RACROUTE REQUEST=FASTAUTH,
 CLASS=LCF,
 ENTITY=command, (8-byte command name)
 WKAREA=workarea, (64-byte work area)
 WORKA=SAF-workarea, (512-byte work area)
 ACEE=acee-ptr (address of ACEE)
LCF DC CL8'XLCFCMD'
```

*Example 11-5. Determine if Transaction Is Executable By User*

This example is written for a multiple user address space. The ACEE= parameter isn't needed for a single user address space.

```

RACROUTE REQUEST=FASTAUTH
 CLASS=FIELD,
 ENTITY=salary, (field name,access,program)
 WKAREA=workarea, (64-byte work area)
 WORKA=SAF-workarea, (512-byte work area)
 ACEE=acee-ptr (address of ACEE)
FIELD DC C'FLD-?RR?' (field; assumed owned)
SALARY DC C'SALARY91',X'60',CL8' '
```

*Example 11-6. Determine if User Has UPDATE Access to Database Field*

### 11.2.1.11 The RACROUTE REQUEST=EXTRACT Call

RACROUTE REQUEST=EXTRACT gives you the ability to extract user information from the Security File. It also encrypts data using either a hashing technique or the DES algorithm. A password encrypted with RACROUTE REQUEST=EXTRACT can be specified as input to RACROUTE REQUEST=VERIFY (ENVIR=CREATE). The RACROUTE REQUEST=EXTRACT call returns an encrypted password for any user whose submitting ACID has the authority to submit jobs on behalf of another user's ACID by issuing the following TSS command function:

```
TSS PER(USER) ACID(OTHER)
```

CA-Top Secret supports a form of extract that returns a feedback area when used in WARN mode if the FIELDS parameter is coded as follows:

```
F'1',C'PASSWORD',C'TSS '
```

This extract indicates whether a user is also authorized to submit a job using another user's ACID.

To issue a RACROUTE REQUEST=EXTRACT call, the caller must be executing authorized—either APF, system key (keys 0 through 7), or supervisor state.

Upon return from a RACROUTE REQUEST=EXTRACT call with the TYPE=EXTRACT parameter, general purpose register 1 points to a response area as documented in the IBM manuals. **It is your responsibility to free this storage area.** The storage is obtained in the subpool specified on the macro invocation, with the default subpool being 229. The storage key obtained depends on the subpool. For subpool definitions, consult the IBM manual: *IBM Debugging Handbook Volume 1*.

**Password Encryption:** RACROUTE REQUEST=EXTRACT is used by the IBM product BDT to obtain an encrypted password as it exists on a RACF file, and passes it over the network to another RACF site where the same user's password exists on the other site's Security File. CA supports this function if the two remote sites have CA-Top Secret and the same encryption key applied using the TSSKEY00 utility.

Using the encrypt function of RACROUTE REQUEST=EXTRACT, password reauthentication is possible. If a password for a user is encrypted using the DES function of RACROUTE REQUEST=EXTRACT, it is encrypted to the same value as would be returned from the RACROUTE REQUEST=EXTRACT extract function.

Consider the following VTAM application example:

1. The user has signed on.
2. At some point in processing reauthentication is required, and the application prompts the user for the password.
3. The password entered by the user is encrypted using:

```
RACROUTE REQUEST=EXTRACT,TYPE=ENCRYPT
```

The DES function encrypts the user's password; assume this is placed in PASSA.

4. The application then issues:

```
RACROUTE REQUEST=EXTRACT,TYPE=EXTRACT
```

which returns the signed-on user's password encrypted; assume this is placed in PASSB.

5. PASSA and PASSB are then compared. If they are the same, then the user is authenticated. (Note that the clear text password isn't returned by RACROUTE REQUEST=EXTRACT.)

**Obtaining Feedback:** To obtain feedback from RACXTRT, code:

```

RACROUTE REQUEST=EXTRACT,
 TYPE=EXTRACT,
 SUBPOOL=1,
 FIELDS=XFIELDS,
 WORKA=RACWK,
 ENTITY=XUSER
XFIELDS DC F'1',C'PASSWORD',C'TSS '
XUSER DC CL8'USERID'

```

**Note:** SUBPOOL=1 obtains storage for response in the user's TCB key. It is your responsibility to free the response area.

For FAIL and IMPLEMENT modes, the return code is 8 if the user isn't authorized, and a value of 9D appears in Register 0. The ACID isn't authorized by CA-Top Secret to extract passwords or to submit jobs on behalf of other users.

In WARN mode, since the value for both Register 15 and Register 0 is 0, the only way to determine if the ACID is authorized to extract passwords and submit jobs on behalf of other ACIDs is through the feedback area. This area follows the encrypted password field mapped by #FEEDBCK in Optional Materials, and #RXTRESP maps the response area. The feedback area is discussed in the next section.

## 11.2.2 Information Feedback

The INSTLN operand of any security macro can be used to obtain information feedback. Feedback consists of return codes, access masks, and message text. It allows a caller to make informative decisions about access attempts and the security environment. The field is a minimum of 16 bytes, a maximum of 255 bytes.

INSTLN must address a data area that is modifiable using the caller's protect key. The format of the feedback area is covered next.

### +0 (4) FEEDBACK ID

Characters TSSF indicate CA-Top Secret user feedback.

**+4 (1) LENGTH** Indicates the size of the feedback area: 16 to 255.

### +5 (1) RETURN CODE

The actual return code; the code that would have been returned to register 15 if the event was processed in FAIL or IMPLEMENT mode. The code will always be returned to the feedback area, even if you are running in WARN or DORMANT mode.

The return codes set in register 15 versus the feedback return codes are:

| R15 | FDBRC | FDBDRC | Meaning            |
|-----|-------|--------|--------------------|
| 0   | 0     | 0      | Access was allowed |

| <b>R15</b> | <b>FDBRC</b> | <b>FDBDRC</b> | <b>Meaning</b>                     |
|------------|--------------|---------------|------------------------------------|
| 4          | 4            | 0             | Resource not defined; DORMANT mode |
| 5          | 4            | -0            | CLASSname not defined; DRC(NOVIOL) |
| 4          | >4           | -0            | WARN mode                          |
| >4         | >4           | -0            | Fail access                        |

For most applications, testing the register 15 code should be sufficient. A return code of 0 allows the request, a return code of 4 defers to whatever native security is available, and a return code greater than 4 fails the request.

In addition, certain fields are set in the 16 word workarea provided for RACROUTE REQUEST=FASTAUTH and FRACHECK:

Word 12 = 0 No audit  
           = 4 User/resource/facility is audited  
 Word 13 = Same as register 15 return code

#### **+6 (1) VIOLATION CODE**

The detail error reason code, 1 to 255, that reflects the type of violation.

#### **+7 (1) REQUESTED ACCESS**

Indicates the type of requested access.

#### **+8 (1) ALLOWED ACCESS**

Indicates the type of access that was granted.

#### **+9 (1) FLAG BYTE**

The FLAG BYTE is used both by the caller, to control processing, and by CA-Top Secret, to feed information back.

The caller can set the NOLOG x'10' bit to prohibit CA-Top Secret from automatically logging the request.

CA-Top Secret sets the terminate user bit, x'08', when the user's violation count has exceeded the VTHRESH threshold. This informs the caller to cancel the session.

The flag byte should be cleared to hexadecimal zeros and set prior to each call because CA-Top Secret will set the flags during each call. If the flag byte isn't reset, incorrect results may occur. The flags are listed next.

| Flag               | Meaning                |
|--------------------|------------------------|
| <b>B'1000000'</b>  | User ACID is undefined |
| <b>B'0100000'</b>  | Default ACID used      |
| <b>B'0010000'</b>  | Password was changed   |
| <b>B'00010000'</b> | Don't log this call    |
| <b>B'00001000'</b> | Terminate this user    |
| <b>B'00000100'</b> | Reserved               |
| <b>B'00000010'</b> | Reserved               |
| <b>B'00000001'</b> | Reserved               |

**+10 (1) USER'S MODE**

The user's MODE is returned by CA-Top Secret. It can be used to determine whether or not to fail the request. The current MODE of the user is returned as:

- x'80' DORMANT
- x'40' WARN
- x'20' FAIL
- x'30' IMPL (10+20)

**+11 (1) MESSAGE COUNT**

Indicates how many messages were returned in the MESSAGE SEGMENTS area; +26.

**+12 (14) RESERVED**

Reserved.

**+26 (?) MESSAGE SEGMENTS**

Message segments of generated messages. The format is: +0(2)=length; +2(?)=message segment. This will contain various messages—including the last-used message for RACINIT. A message segment example appears below.

- +26(2) = 53
- +28(53) = TSS9500E DUF/EXTRACT FAILED-USER HAS NO (INST) DATA
- +81(2) = 42
- +83(44) = TSS9506E PROBABLE SITE INTERFACING ERROR 030

**Note:** All lengths and offsets are in decimal, not hexadecimal, format.

Message text is only returned if the feedback area size will hold the text. RACROUTE REQUEST=FASTAUTH supports a feedback area of 256 bytes in length.

### 11.2.3 Multiple User Address Space

This section contains information regarding the CA-Top Secret environment and explains the concept and use of the ACEE parameter. The ACEE must be passed when customized security checks are written for use in multiple user address space systems such as CICS, DL/1, CA-Roscoe, and CA-IDMS.

### 11.2.3.1 The CA-Top Secret Environment

CA-Top Secret maintains a security environment within each and every active address space. This environment is created at job/session initiation through a call to CA-Top Secret. VSE automatically invokes initiation security for BATCH, TSO, Started Task, DL/1, CICS, and NCCF facilities. Any installation facility can use the same approach in creating the security environment.

The environment consists of control blocks initialized and maintained solely by CA-Top Secret. The major control block in this environment is called the ACEE, Accessor Environment Element. It is recognized by VSE as a standard security control block. The site doesn't have to be concerned about the content of the ACEE; what **is** important is the need to associate individual users within the local facility with their unique ACEEs.

The information stored within the ACEE is used by CA-Top Secret to validate resource access within the address space for an active user. In single user facilities, the system invokes security to build and maintain a single Master ACEE. This is also true of multi-user facilities, so a Master ACEE will always exist.

For CA-Top Secret, the Master ACEE is associated with the Accessor ID (ACID) for the BATCH job that is the facility. The facility is responsible for invoking CA-Top Secret (by issuing RACROUTE REQUEST=VERIFY) to build ACEEs for individual users as they sign on.

Each individual ACEE reflects the security-related information for the user's ACID. When resource validation takes place in any address space, CA-Top Secret uses the information associated with either the Master ACEE or the user ACEE to validate access to the resource. In most cases, the security driver that invokes CA-Top Secret is VSE. However, other forms of resource checking must be initiated by the installation facility.

However, when an address space contains multiple users (a multiple user address space), if there is no task-per-user relationship (each TCB anchors its own ACEE), then the ACEE that will most often be used is the Master ACEE and not the individual user's ACEE. In this case, the installation code should always provide ACEE= on all resource calls to perform security validation for each user.

**Note:** When a subtask is created by VSE through the ATTACH macro, the TCBSENV field of the mother task isn't propagated to the daughter task. If it is necessary for the daughter task to retain the same ACEE (or ACID information), the mother task must fill in the daughter task's TCBSENV field.

As an example of a no task-per-user environment, an installation-written facility has its own control block that reflects individual user information. In this example, the installation control block is called the USER C/B.

The USER C/B must contain the address of the ACEE for the user. This address is passed to security for all future security checks. All facilities, whether or not they are VSE drivers or installation code, interface with CA-Top Secret through the following standard VSE macro instructions.

To perform signon security and build the ACEE:

```
RACROUTE REQUEST=VERIFY,ENVIR=CREATE
```

To perform signoff cleanup and free the ACEE:

```
RACROUTE REQUEST=VERIFY,ENVIR=DELETE
```

To perform data set, volume, terminal, abstract, and application checking:

```
RACROUTE REQUEST=AUTH
```

To perform LCF checking for transactions and commands:

```
RACROUTE REQUEST=FASTAUTH
```



The ACEE keyword for these macros is the key to multi-user security validation. It passes the address of the ACEE associated with the user prior to actually performing the requested operation (such as opening a data set or calling a module). If the ACEE isn't passed, VSE uses the Master ACEE to validate the request. Although this is valid, it doesn't perform security validation for the user. Security validation must be performed prior to allowing the following operations to take place:

1. Opening a data set
2. Creating, scratching, or renaming a data set
3. Executing a transaction or command
4. Submitting a jobstream.

*For security and integrity reasons, Computer Associates will NOT distribute any source macro listings of CA-Top Secret control blocks.*

### 11.2.3.2 Programming Examples

The following samples illustrate the technique required for providing security for an installation's multi-user facility.

**Note:** The data areas for this programming example can be found on 11.2.2, "Information Feedback" on page 11-36.

The following scenario is used for the samples:

- Process signon by the user.

To determine if a user is defined to CA-Top Secret, you must use the information feedback mechanism at signon processing. A flag indicates whether or not the user is defined or if a default ACID was used. Refer to 11.2.2, "Information Feedback" on page 11-36 for details.

- All user-related information is stored in the USER C/B control block. Here the use of the ACID, the password, and the terminal are verified.
- If other data is required for any installation exit code (such as accounting data), it can also be passed. RACROUTE REQUEST=VERIFY provides this checking.

**Note:** A return code of zero will normally be returned in WARN or DORMANT modes.

## 11.2 Extending Security Through the VSE Security Interface

|          |                                                                                                                                                   |                                                                                                                                                  |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| L        | R9,CURRENT                                                                                                                                        | CURRENT USER                                                                                                                                     |
| USING    | USERCB,R9                                                                                                                                         | ADDRESS OF THIS USER'S C/B                                                                                                                       |
| LA       | R2,UCBTERM                                                                                                                                        | TERMINAL NAME                                                                                                                                    |
| LA       | R3,UCBUID                                                                                                                                         | USER ID                                                                                                                                          |
| LA       | R4,UCBPASSW                                                                                                                                       | PASSWORD                                                                                                                                         |
| LA       | R6,UCBNEWPW                                                                                                                                       | NEW PASSWORD IF ANY                                                                                                                              |
| LA       | R9,UCBACEE                                                                                                                                        | ACEE POINTER                                                                                                                                     |
| MVC      | RVFY,XRVFY                                                                                                                                        | TRANSFER MF=L SKELETON                                                                                                                           |
| RACROUTE | REQUEST=VERIFY,<br>ENVIR=CREATE,<br>WORKA=SAFWORK,<br>USERID=(R3),<br>PASSWRD=(R4),<br>NEWPASS=(R6),<br>TERMID=(R2),<br>ACEE=(R9),<br>MF=(E,RVfy) | 512 BYTE SAF WORK AREA<br>VALIDATE USERID AS ACID<br>AND PASSWORD<br>ALLOW NEW PASSWORD<br>TERMINAL CHECK<br>THIS WILL HOLD ACEE PTR UPON RETURN |
| LTR      | R15,R15                                                                                                                                           | SUCCESSFUL INITIATION?                                                                                                                           |
| BNZ      | INITERRI                                                                                                                                          | BR NO...PERFORM ERROR PROCESSING                                                                                                                 |

*Example 11-8. Verify Use of ACID, Password, and Terminal*

At this point, UCBACEE now contains the address of the ACEE that was created for the user. This ACEE must always be passed whenever security validation is performed for the user.

|          |                                                                                                                                                               |                                                                                                                                          |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| L        | R9,CURRENT                                                                                                                                                    | ACTIVE USER C/B                                                                                                                          |
| L        | R2,UCBACEE                                                                                                                                                    | ACEE FOR THIS USER                                                                                                                       |
| IC       | R0,ACCESS                                                                                                                                                     | REQUESTED ACCESS LEVEL                                                                                                                   |
| MVC      | RAUTH,XRAUTH                                                                                                                                                  | TRANSFER SKELETON                                                                                                                        |
| RACROUTE | REQUEST=AUTH,<br>CLASS='DATASET',<br>WORKA=SAFWORK,<br>ENTITY=DSNAME,<br>VOLSER=VOLUME,<br>DSTYPE=N,<br>ATTR=(R0),<br>LOG=NONE,<br>ACEE=(R2),<br>MF=(E,RAUTH) | DATA SET CHECK<br>512 BYTE WORK AREA<br>WITH DATA SET NAME<br>AND VOLUME<br>NON-VSAM<br>ACCESS MASK<br>IGNORE LOGGING<br>FOR ACTIVE USER |
| LTR      | R15,R15                                                                                                                                                       | IS USER AUTHORIZED?                                                                                                                      |
| BNZ      | DSNERR1                                                                                                                                                       | IF NOT AUTHORIZED, BRANCH                                                                                                                |

*Example 11-9. Check for Authority to Open Data Set*

|          |                                                                                                    |                                                                                         |
|----------|----------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------|
| L        | R9,CURRENT                                                                                         | ACTIVE USER C/B                                                                         |
| L        | R2,UCBACEE                                                                                         | ACEE FOR THIS USER                                                                      |
| MVC      | RAUTH,XRAUTH                                                                                       | TRANSFER SKELETON                                                                       |
| RACROUTE | REQUEST=AUTH,<br>CLASS='APPL',<br>WORKA=SAFWORK,<br>ENTITY=APPLICTN,<br>ACEE=(R2),<br>MF=(E,RAUTH) | APPLICATION CHECK<br>512 BYTE WORK AREA<br>FOR REQUESTED APPLICATION<br>FOR ACTIVE USER |
| LTR      | R15,R15                                                                                            | IS USER AUTHORIZED?                                                                     |
| BNZ      | APPLERR1                                                                                           | IF NOT AUTHORIZED, BRANCH                                                               |

*Example 11-10. Check for Authority to Use Application*

|          |                                                                                                     |                                                                                 |
|----------|-----------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------|
| L        | R9,CURRENT                                                                                          | ACTIVE USER C/B                                                                 |
| L        | R2,UCBACEE                                                                                          | ACEE FOR THIS USER                                                              |
| MVC      | RFAUTH,XRFAUTH                                                                                      | TRANSFER SKELETON                                                               |
| RACROUTE | REQUEST=FASTAUTH,<br>CLASS=LCF,<br>WORKA=SAFWORK,<br>WKAREA=FAUTHWK,<br>ACEE=(R2),<br>MF=(E,RFAUTH) | LCF CHECK<br>512 BYTE WORK AREA<br>PASS 16 WORD WORK AREA<br>ACTIVE USER'S ACEE |
| LTR      | R15,R15                                                                                             | DID USER PASS VALIDATION?                                                       |
| BNZ      | EXECERR1                                                                                            | CAN NOT EXECUTE MODULE, BRANCH                                                  |

*Example 11-11. Check for Authority to Execute Transaction/Command*

|          |                                                                                                                          |                                   |
|----------|--------------------------------------------------------------------------------------------------------------------------|-----------------------------------|
| RACROUTE | REQUEST=FASTAUTH,<br>CLASS=USER5,<br>WORKA=SAFWORK,<br>ENTITY=SPECIAL,<br>WKAREA=FAUTHWK,<br>ACEE=(R2),<br>MF=(E,RFAUTH) | USER5 CHECK<br>512 BYTE WORK AREA |
|----------|--------------------------------------------------------------------------------------------------------------------------|-----------------------------------|

*Example 11-12. Check Authority to Access a User Resource*

|          |                                                              |                    |
|----------|--------------------------------------------------------------|--------------------|
| L        | R9,CURRENT                                                   | ACTIVE USER C/B    |
| LA       | R2,UCBACEE                                                   | ACEE FOR THIS USER |
| MVC      | RVFY,XRVFYD                                                  | TRANSFER SKELETON  |
| RACROUTE | REQUEST=VERIFY,<br>ENVIR=DELETE,<br>ACEE=(R2)<br>MF=(E,RVFY) |                    |

Example 11-13. Process User's Signoff

### 11.2.3.3 Data Areas Used by Sample Programs

|                                                                       |          |                                  |                                    |
|-----------------------------------------------------------------------|----------|----------------------------------|------------------------------------|
| USER5                                                                 | DC       | CL8'USER5'                       | Class name for FASTAUTH            |
| * special contains user resource 10000, update access level, program: |          |                                  |                                    |
| SPECIAL                                                               | DC       | CL8'10000',X'60',CL8'CUTRPGM'    |                                    |
| LCF                                                                   | DC       | CL8'XLCF'                        |                                    |
| XRVFY                                                                 | RACROUTE | REQUEST=VERIFY,ENVIR=CREATE,MF=L |                                    |
| XRVFYD                                                                | RACROUTE | REQUEST=VERIFY,ENVIR=DELETE,MF=L |                                    |
| XRAUTH                                                                | RACROUTE | REQUEST=AUTH,MF=L                |                                    |
| XRFAUTH                                                               | RACROUTE | REQUEST=FASTAUTH,MF=L            |                                    |
| USERCB                                                                | DSECT    |                                  | USER CONTROL BLOCK MAPPING         |
| UCBUID                                                                | DS       | XL1,CL8                          | USER LENGTH AND USERID             |
| UCBPASSW                                                              | DS       | XL1,CL8                          | PASSWORD LENGTH AND PASSWORD       |
| UCBNEWPW                                                              | DS       | XL1,CL8                          | NEW PASSWORD LENGTH AND PASSWORD   |
| UCBTERM                                                               | DS       | CL8                              | TERMINAL USER IS USING             |
| UCBACEE                                                               | DS       | A                                | POINTER TO USER'S ACEE             |
| WORKAREA                                                              | DSECT    |                                  |                                    |
| CURRENT                                                               | DS       | A                                | ADDRESS OF ACTIVE USER C/B         |
| FAUTHWK                                                               | DS       | 16F                              | WORKAREA FOR FASTAUTH              |
| SAFWORK                                                               | DS       | XL512                            | SAF WORK AREA                      |
| RVFY                                                                  | DS       | 0XL256                           | RACROUTE PARMLIST FOR EXECUTION    |
| RAUTH                                                                 | DS       | 0XL256                           | " " " " " " " " " " " " " "        |
| RFAUTH                                                                | DS       | XL256                            | " " " " " " " " " " " " " "        |
| XACTNNAM                                                              | DS       | CL8                              | NAME OF MODULE TO BE EXECUTED      |
| APPLCTN                                                               | DS       | CL8                              | NAME OF APPLICATION TO BE ACCESSED |
| DSNAME                                                                | DS       | CL44                             | NAME OF DATA SET TO BE ACCESSED    |
| ACCESS                                                                | DS       | X                                | ACCESS LEVEL FOR DATA SET OPEN     |
| VOLUME                                                                | DS       | CL6                              | VOLUME FOR DATA SET                |

## 11.2.4 Resource Name List Service Support

CA-Top Secret Release 3.0 and above provide Resource Name List support. This support allows authorized programs to call CA-Top Secret to retrieve the names of resources within a class that a given ACID can access at READ level or higher. This support is provided by CA-Top Secret module TSSRSVC1.

To perform this function, TSSRSVC1 searches the RDT for the class name. If the class is found, TSSRSVC1 checks all permissions and determines if the specified ACID is authorized to access the resource at READ level or higher. When TSSRSVC1 finds a match, it places the resource name into the input work area. CA-Top Secret supports any **general resource** or **PIE resource** (resource classes that support masking) in the RDT.

CA-Top Secret searches the USER, PROFILE, and ALL records, but may stop if the size of the output list exceeds the work area.

CA-Top Secret loads the address of TSSRSVC1 into RCVTPNL0 during initialization. The caller of the module must use the address in RCVTPNL0 in the RCVT. The RCVT is mapped by the IBM ICHPRCVT macro.

Note the following items:

- Your program is responsible for obtaining and releasing the storage that TSSRSVC1 uses to store the Resource Name List.
- Callers of TSSRSVC1 must be running in key 0, task mode, with no locks held.

### 11.2.4.1 Invoking Support

When you invoke TSSRSVC1 support, your program must pass the following four parameters to it:

|                         |                                                                                                                            |
|-------------------------|----------------------------------------------------------------------------------------------------------------------------|
| <b>Classname</b>        | An eight-character class name from which CA-Top Secret derives the resource names to which the ACID has authorization.     |
| <b>Work Area Length</b> | A fullword that contains the length of the area in which TSSRSVC1 is going to build the Resource Name List.                |
| <b>Work Area</b>        | A fullword pointer that contains the address of the workarea where TSSRSVC1 is going to build the Resource Name List.      |
| <b>ACEE Pointer</b>     | A fullword pointer that contains the address of the ACEE for the ACID for whom resource authorization is being determined. |

The calling program passes these parameters to TSSRSVC1 using the CALL command.

### 11.2.4.2 Returned Format

A fixed (31) count field that precedes the Resource Name List contains the total count of resource names returned by TSSRSVC1. The format of the list when it is placed in the work area is:

**Name Length** The two byte length of the resource name.

**Flags** A one byte flag field (only the first bit is used).

**Note:** The first bit of the flag field byte is on if the resource is a generic resource.

**Resource Name** A variable length resource name.

### 11.2.4.3 Return Codes

The return codes from TSSRSVC1 follow CA-Top Secret conventions with a return code of 0 indicating a successful search.

The following return codes are returned in register 15 and the reason codes are returned in register 0. (All return and reason codes are shown in hexadecimal.)

| <b>Code</b> | <b>Meaning</b>                                                                                     |
|-------------|----------------------------------------------------------------------------------------------------|
| <b>00</b>   | The Resource Name List function completed successfully.                                            |
| <b>04</b>   | No resources found for which the ACID had at least READ access.                                    |
| <b>08</b>   | No resource entries found for that class; indicates that no resources existed for the input class. |
| <b>0C</b>   | The work area wasn't large enough to hold all the resource names.                                  |
| <b>14</b>   | Resource Name List parameter error:                                                                |
|             | <b>Reason Code      Meaning</b>                                                                    |
|             | <b>04</b> No ACEE available.                                                                       |
|             | <b>08</b> Work Area too small to contain a single resource.                                        |
|             | <b>10</b> Input Class name not valid.                                                              |

## 11.3 Extending Security Through Site Security Exits

CA-Top Secret allows a site to create customized security checks which can bypass, replace, or enhance normal security validation. By allowing an installation to process requests prior to CA-Top Secret's validation, customized enhancements serve the individual needs of any organization. Moreover, CA-Top Secret makes customizing easy through the use of the Installation Exit, TSSINSTX.

## 11.3.1 Customization Ideas

This chapter contains a list of possible uses for TSSINSTX. Sample code for the uses that are listed below can be found on the CA-Top Secret distribution tape in the TSSOPMAT file member TSSINST1.

- Provide additional job card parameter validations:
  - JES2 Initiator class authorizations
  - Job priority authorizations
  - Account number information
- Limit TSO usage by department.
- Maintain CA-Roscoe usage statistics based on the time-of-day session-Roscoe/ \* \* signoff.
- Voice/image verification for online session signon.
- Provide implicit data set prefix security for DASD management archiving data sets.
- Use "pseudo data set names" to provide other resources with the flexibility of data set name security (that is, character masking, program pathing, and so on). This approach can be useful in controlling access to members in a library management package (for example, CA-Panvalet, CA-Librarian, or CA-Endeavor).
- Eliminate the logging of BYPASS events as desired by the installation to reduce ATF logging.

## 11.3.2 Mechanics

A skeleton for TSSINSTX is supplied on the CA-Top Secret distribution tape in the TSSOPMAT file member TSSINSTX. An activation matrix at the beginning of the TSSINSTX module defines which exit points are invoked. The matrix contains a one byte flag for each function (point of user entry). If the flag byte is non-zero, CA-Top Secret will call the installation exit point for the function.

It is the responsibility of the site programmer to place the customized installation code in the appropriate exit routine within TSSINSTX.

A brief description of the functional requirements of each available entry point appears next.

### 11.3.2.1 Common Exit Parameters

The following list contains common parameters that are passed to all exit points. Some of the parameter fields may not be valid at some exit points depending upon the point in processing at which the exit is invoked. For example, the TXA#DRC parameter field will not contain valid information for the PREINIT exit point since the security processing has not yet completed.

**TXA#TYPE**      @ Generic job type (type=value from Facility Matrix Table)



|                 |                                                              |
|-----------------|--------------------------------------------------------------|
| <b>TXA#ACID</b> | @ ACID that initiated the security event                     |
| <b>TXA#JOB</b>  | @ Jobname that initiated the security event                  |
| <b>TXA#TERM</b> | @ Terminal/source for the ACID initiating the security event |
| <b>TXA#INST</b> | @ Installation-wide installation data field (eight bytes)    |
| <b>TXA#FEED</b> | @ Feedback area address, if present                          |
| <b>TXA#FACM</b> | @ Facility matrix entry for the initiating ACID              |
| <b>TXA#PGMS</b> | @ Initiating programs for this event (from PRB)              |
| <b>TXA#DRC</b>  | @ Detailed Reason Code for this security event               |
| <b>TXA#MODE</b> | @ Mode byte for this event                                   |
| <b>TXA#FLAG</b> | @ Flag for communication with TSSINSTX                       |
| <b>TXA#ACEE</b> | @ ACEE for the ACID that initiated the security event        |
| <b>TXA#SREC</b> | @ SECREC for the ACID that initiated the security event      |
| <b>TXA#SVCS</b> | @ SVCs in control when the security event was initiated      |
| <b>TXA#INSD</b> | @ User installation data                                     |
| <b>TXAXINSD</b> | @ User installation data                                     |
| <b>TXAXLANG</b> | @ Language indicator                                         |
| <b>TXAXINSW</b> | @ Installation field that can be used with the ACEE          |

### 11.3.2.2 PREINIT

Receives control on all initiation requests after the initial parameter list validation, but prior to processing.

#### Exit Point Specific Parameters:

|                 |                                                                                                                                                                                     |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>TXAIUSER</b> | @ Userid (ACID) being signed on                                                                                                                                                     |
| <b>XAIPGMR</b>  | @ Programmer name field (20 chars, batch only)                                                                                                                                      |
| <b>TXAIACCT</b> | @ Accounting information (batch only)<br>+0 = 1 byte. Number of fields following this field format:<br>+0 = 1 byte length (0 implies null positional field)<br>+1 = accounting data |
| <b>TXAIPASS</b> | @ Old password followed by new password as eight character blank padded fields                                                                                                      |

#### Exit Point Specific Output:

|                  |                                               |
|------------------|-----------------------------------------------|
| <b>\$TXAIIMC</b> | Flag indicating change in user mode           |
| <b>\$TXAIAUD</b> | Flag indicating that user should be audited   |
| <b>\$TXAISUS</b> | Flag indicating that user should be suspended |

**Return Codes:**

|                  |                                    |
|------------------|------------------------------------|
| <b>00</b>        | Normal continuance                 |
| <b>04</b>        | Fail request                       |
| <b>08</b>        | Continue without further checking  |
| <b>0c</b>        | Continue in BYPASS mode            |
| <b>10</b>        | Reserved                           |
| <b>14</b>        | Continue without password checking |
| <b>any other</b> | Fail request with RC=28 DRC=02     |

### 11.3.2.3 VOLUME

Receives control on all volume checks allowing additional user validation.

**Exit Point Specific Parameters:**

|                 |                                         |
|-----------------|-----------------------------------------|
| <b>TXA#VOL</b>  | @ VOLSER (six bytes, blank padded)      |
| <b>TXA#DSN</b>  | @ DATASET NAME (44 bytes, blank padded) |
| <b>TXA#ACC</b>  | @ Requested access level (2 bytes)      |
| <b>TXA#RTYP</b> | @ Resource type (rescode from RDT)      |

**Exit Point Specific Output:**

|                |                |
|----------------|----------------|
| <b>TXA#VOL</b> | May be changed |
| <b>TXA#DSN</b> | May be changed |
| <b>TXA#ACC</b> | May be changed |

**Return Codes:**

|           |                                   |
|-----------|-----------------------------------|
| <b>00</b> | Normal continuance                |
| <b>04</b> | Fail request                      |
| <b>08</b> | Continue without further checking |

### 11.3.2.4 DATASET

Receives control on all data set checks allowing additional user validation.

**Exit Point Specific Parameters:**

|                 |                                         |
|-----------------|-----------------------------------------|
| <b>TXA#DSN</b>  | @ DATASET NAME (44 bytes, blank padded) |
| <b>TXA#VOL</b>  | @ VOLSER (six bytes, blank padded)      |
| <b>TXA#ACC</b>  | @ Requested access level (two bytes)    |
| <b>TXA#RTYP</b> | @ Resource type (rescode from RDT)      |

**Exit Point Specific Output:**

TXA#DSN May be changed

TXA#VOL May be changed

TXA#ACC May be changed

**Return Codes:**

00 Normal continuance

04 Fail request

08 Continue without further checking

**11.3.2.5 RESOURCE**

Receives control on resource checks for classes with the EXIT attribute in the RDT, except DATASET and VOLUME. Resources with no RDT entry (that is, LCF) do not go through this exit point.

**Exit Point Specific Parameters:**

TXA#RESN @ Resource name (ENTITY from RACFPL)

TXA#ACC @ Requested access level (two bytes)

TXA#RTYP @ Resource type (rescode from RDT)

**Exit Point Specific Output:**

TXA#RESN May be changed

TXA#RTYP May be changed

TXA#ACC May be changed

**Return Codes:**

00 Normal continuance

04 Fail request

08 Continue without further checking

**11.3.2.6 COMMAND**

Receives control on each TSS command.

**Exit Point Specific Parameters:**

TXACCACI @ Caller's ACID

TXACTYPE @ Caller's TYPE

TXACMRTN @ Message routine called by passing WTO style buffer to the routine

**TXACCBUF** @ Command buffer

**TXACFUNC** @ TSS command function

01 = CREATE  
 02 = DELETE  
 03 = ADD  
 04 = REPLACE  
 05 = RENAME  
 06 = REMOVE  
 07 = PERMIT  
 08 = REVOKE  
 09 = WHOOWNS  
 0A = WHOHAS  
 0B = LIST  
 0C = HELP  
 0D = LOCK  
 0E = UNLOCK  
 0F = WHOAMI  
 10 = MODIFY  
 11 = ADMIN  
 12 = DEADMIN  
 13 = MOVE

**TXACACID** @ ACID being administered

**TXACPROF** @ Profile table

**TXACACIL** @ Permitted ACIDs table

**TXACPASS** @ Password data

**TXACOPID** @ CICS OPIDENT

**TXACINST** @ Installation data

**TXACAUD** @ Audit flag

**TXACTGTS** @ Target list or sender name

**TXACRLST** @ Resource list. List of fullwords each pointing to a resource table.  
 Use fullword '0' for delimiter.

**Exit Point Specific Output:** NONE

**Return Codes:**

**00** Normal continuance

**non-zero** FAIL command

**Note:** The common exit parameters are not passed to COMMAND.

This exit point is not protected by the same ESTAE routine that protects most other entry points, and is not disabled automatically when an abend occurs.

### 11.3.2.7 TERM

Receives control on termination requests that resulted from an out-of-address space termination.

**Exit Point Specific Parameters:** NONE

**Exit Point Specific Output:** NONE

**Return Codes:**

- 00** Normal continuance
- 04** Fail request
- 08** Continue without further checking

### 11.3.2.8 POSTINIT

Receives control on initiation requests after all processing and prior to returning control to the caller.

**Exit Point Specific Parameters:**

- TXAINAME** @ ACID name data
- TXAIUSER** @ USERID
- TXAIPGMR** @ Programmer name
- TXAIACCT** @ Accounting data
- TXAIPASS** @ Password

**Exit Point Specific Output:**

- \$TXAIIMC** Change in user mode
- \$TXAIAUD** User should be audited

**Return Codes:**

- 00** Normal continuance
- 04** Fail request
- 08** Continue without further checking

### 11.3.2.9 UNDEFIND

Receives control on initiation requests for undefined users prior to returning control to the caller.

**Exit Point Specific Parameters:**

**TXAIUSER** @ USERID  
**TXAIPGMR** @ Programmer name  
**TXAIACCT** @ Accounting data  
**TXAIPASS** @ Password

**Exit Point Specific Output:**

May alter TXA#ACID, TXA#JOB, TXA#TERM or TXA#IPASS

**Return Codes:**

**10** Retry with new ACID

### 11.3.2.10 PASSWORD

Receives control of security requests for which a password change has been requested.

**Exit Point Specific Parameters**

**TXAIUSER** @ USERID  
**TXAIPGMR** @ Programmer name  
**TXAIACCT** @ Accounting data  
**TXAIPASS** @ Password  
**TXAINAME** @ ACID name field

**Exit Point Specific Output:** NONE

**Return Codes:**

**non-zero** Fail the request

### 11.3.2.11 IO

Receives control on security initiation requests to allow the processing of special device considerations.

**Exit Point Specific Parameters:**

**TXAIPKL** @ PHYSKEY length  
**TXAIPKY** @ PHYSKEY data

**TXAIVBFR** @ Voice buffer (filled by exit code)

**Exit Point Specific Output:**

If voice verification is in use, then the voice buffer must be filled. If voice verification is returned, then the voice buffer must match the user's Security Record.

**Return Codes:**

**00** Normal continuance  
**04** Fail request  
**14** Skip password check

### 11.3.2.12 SESSEND

Receives control of individual user termination requests from multi-user address spaces.

**Exit Point Specific Parameters**

**TXAIUSER** @ USERID  
**TXAIPGMR** @ Programmer name  
**TXAIACCT** @ Accounting data  
**TXAIPASS** @ Password  
**TXAINAME** @ ACID name field

**Exit Point Specific Output:**

**\$TXAIIMC** Change in user mode  
**\$TXAIAUD** Audit this user

**Return Codes:** Not checked

### 11.3.2.13 SUBMIT

Receives control when the job is submitted through the internal reader.

**Exit Point Specific Parameters:**

**TXASSJOB** @ Submitted jobname  
**TXASSACT** @ Submitted accounting data  
**TXASSPGN** @ Submitted programmer name field  
**TXASSACD** @ Submitted ACID  
**TXASAFLG** @ Submit flag byte  
**80** = USER= found on job statement  
**40** = ACID derived from (J,x)

- 20** = ACID propagated
- 10** = PASSWORD= found on job statement
- 08** = NOSUBCHK caller

**Exit Point Specific Output:** NONE

**Return Codes:**

- 00** Normal continuance
- 08** Submit without further checking
- any other** Fail the Submit

### 11.3.2.14 CHANGE

Receives control for all changes that update the Recovery File.

**Exit Point Specific Parameters:**

- TXA#RBUF** @ Recovery file buffer
- +0 = x'11'** Constant
  - +1 = h** Length of data
  - +3 = c11** p=password change, u=DUFUPD, c=TSS command
  - +4 = c18** ACID
  - +c = x13** Owning ACID number
  - +f = p13** Date in the format yymmmF
  - +12 = x14** Time in timer units
  - +16 = c14** CPUID
  - +1a = x11** TSS version indicator
  - +1b = x13** ACID number
  - +22 = variable** Depending upon call:
    - c18** Encrypted password for password change
    - or**
    - h** Length of data (including this field)
    - h** Reserved
    - clxxx** TSS command text
- TXA#CPL** @ Change parameter list
- +0 = c11** p=password change, u=DUFUPD, c=TSS command
  - +F = c18** ACID
  - +17 = c18** Owning ACID



**Exit Point Specific Output:** NONE

**Return Codes:** Not checked

**Note:** This exit point receives control on:

1. Password change
2. DUFUPD
3. TSS command Security File change

**Note:** When entered for a TSS command processor change, this exit point is protected by the CA-Top Secret Command Processor ESTAE routine. Abends will be recovered, but the exit will not be disabled.

### 11.3.2.15 ACTION

Receives control for all security events against DATASETS, VOLUMES and masked prefixed resources that match on a PERMIT with ACTION(EXIT).

**Exit Point Specific Parameters:**

**TXA#XE** @ Internally formatted resource permission

**Exit Point Specific Output:** NONE

**Return Codes**

**00** Normal continuance

**non-zero** Fail the request

### 11.3.2.16 MESSAGE

Receives control for security messages. This allows the user to edit the text.

**Exit Point Specific Parameters:**

**TXA#MBUF** @ Message buffer in WTO format

**Exit Point Specific Output:**

**TXA#MBUF** May alter message text

**Return Codes:** NONE

### 11.3.2.17 VIOLATN

Receives control for all logged security events. This allows the user to alter normal logging functions.

**Exit Point Specific Parameters:**

**TXA#FLOG** @ FLOG buffer

**TXA#DRCE** @ Detailed reason code element

**TXA#3** @ Audit flag

**80** Bypass

**20** DSN/VOL/RESOURCE exit allowed access

**10** Bypass DSN check(NODSNCHK)

**04** TEMP DSN and TEMPDSN(NO) set

**Note:** These flags are not mutually exclusive. Any combination of flags may be set.

**Exit Point Specific Output:** NONE

**Return Codes:**

**00** Normal continuance

**04** Don't log this call

**11.3.2.18 SITE**

Site exit point, through RACROUTE REQUEST=AUTH,CLASS=INSTEXIT, that allows for user processing.

**Exit Point Specific Parameters:**

**TXA#DATA** @ ENTITY= from RACROUTE

**Exit Point Specific Output:** NONE

**Return Codes:**

**00** Normal continuance

**11.3.2.19 CPF**

Allows the user to examine CPF propagation requests and change the target machine(s).

**Exit Point Specific Parameters:**

**TXACTGTS** @ CPF target data

**TXACCACI** @ Issuing ACID

**TXACTYPE** @ Issuing ACID's type

**TXACCBUF** @ Raw command buffer

**TXACFUNC** @ Command function

**Exit Point Specific Output:** May alter CPF target names or eliminate targets by altering \$CTFLG1 and \$CTUSE

**Return Codes:**

**00** Normal continuance

**04** Reject the entire command

### 11.3.2.20 TSSINSTX Characteristics

TSSINSTX is a single load module with a single entry point. There are 18 different processing routines which are entered based upon the function code passed on entry. The load module must reside in a library accessible for all partitions, and must be named TSSINSTX. The Link-edit must specify AMODE(31), RMODE(ANY).

Please see sample member TSSINST1.A delivered with CA-Top Secret

TSSINSTX normally runs in the user address space under the security SVC. The exit is entered in supervisor state, key 0. The exit can issue any SVC and perform I/O unless otherwise noted in the list above.

TSSINSTX is protected by an error recovery routine (in most cases). In the event of an abend, an SVC dump is taken and the exit is disabled with a message issued to the security and master consoles. Any variations to this rule are noted in the list above.

CA-Top Secret loads the installation exit (TSSINSTX) as specified on the module's linkedit attributes. The installation exit should be linked as RMODE(ANY), so that it will be loaded above the line. Because most parameters passed to the exit now exist above the line, RACROUTE calls must be issued for all security checks. TSSINST1.Z supplied in the CA-Top Secret Installation Library, contains examples of proper coding of RACROUTE requests.

### 11.3.3 Activating the Installation Exit

The CA-Top Secret control option, EXIT, controls the activation or deactivation of the installation exit. The exit may be activated or deactivated at any time.

**EXIT(ON)** Activates the installation exit module, TSSINSTX. If the EXIT option is not specified, then the default is EXIT(OFF). If EXIT(ON) is specified but a copy of TSSINSTX is not found (in the LINKLIST), CA-Top Secret ignores this control option.

**EXIT(OFF)** Deactivates the installation exit module, TSSINSTX.

A new exit can be activated at any time (without bringing down the CA-Top Secret started task) using the following procedures:

1. TSS MODIFY(EXIT(OFF)) (if required)
2. Reassemble and link TSSINSTX into the linklst
3. TSS MODIFY(EXIT(ON))

### 11.3.4 CA-Top Secret Exit/User Entry Points

TSSINSTX offers 18 points where you can exit CA-Top Secret and enter your site's customized security check code; these points are listed below.

Member TSSOPMAT on the distribution tape contains the TSSINSTX module and sample installation exit code in member TSSINST1.

| Entry Label     | Code | Description                          |
|-----------------|------|--------------------------------------|
| PREINIT         | 00   | Job/session pre-initiation           |
| VOLUME          | 04   | Volume access validation             |
| DATASET         | 08   | Data set access validation           |
| RESOURCE        | 12   | General resource validation          |
| COMMAND         | 16   | TSS command use validation           |
| TERM            | 20   | Job (Address Space) termination      |
| POSTINIT        | 24   | Job/session initiation completion    |
| UNDEFIND        | 28   | Undefined ACID entry                 |
| PASSWORD        | 32   | Password change validation           |
| I/O             | 36   | Voice/image and special terminal I/O |
| SESSEND         | 40   | Individual session termination       |
| SUBMIT          | 44   | INTRDR job submission                |
| CHANGE          | 48   | Security File change                 |
| ACTION          | 52   | Permit action exit                   |
| MESSAGE         | 56   | CA-Top Secret message editing        |
| VIOLATN/LOGGING | 64   | Violations and all audited events    |
| RESERVED        | 68   | Reserved for future use              |
| CPF             | 72   | Command propagation exit             |

**Note:** Although the text of the CA-Top Secret messages can be changed, the numbers can't be edited.

# Appendix A. Computer Operations Guide

---

This chapter outlines the procedures most commonly used by the operations department to start and maintain the CA-Top Secret security product. It is intended as a sample guide to be used as a base document by the security administrator. The security administrator should determine which elements of the information presented are applicable to the needs of the operation staff and edit this sample accordingly. Further, this sample should be "customized" to reflect your site's particular security requirements and organizational structure.

**Note:** You may want to limit your operations staff to only those functions not requiring console authority, since this authority allows a great deal of control over the security environment.

Moreover, the sample guide is intentionally comprehensive in its scope so that potentially relevant topics are not overlooked.

The intended audience for this guide includes: the security administrator responsible for planning the CA-Top Secret implementation, the systems programmer involved with the CA-Top Secret installation, and the operations staff.

Familiarity with and knowledge of the following CA-Top Secret documentation is a prerequisite:

- *Installation*
- *Implementation: of subsystems*
- *Control Options*
- *Planning*
- *Messages and Codes*

CA-Top Secret is an access control product designed for VSE operating systems. For the most part, the operations staff will not interact with CA-Top Secret in any way. There will, however, be a limited number of simple, well-defined system and security maintenance procedures for which the operations staff is primarily responsible.

The following list outlines the areas of operator responsibility discussed in this chapter.

- Changing Control Options
- Entering Commands
- Startup and Shutdown Procedures
- Backup Procedures
- Restore Procedures
- Operator Accountability
- Messages and Codes



## A.1 Control Options

Using CA-Top Secret control options is one way security administrators control security. It is recommended that only an MSCA or SCA be allowed to select and change control options. There are, however, conditions under which your security administrator may elect to allow an operations supervisor to enter control options.

CA-Top Secret protects the most powerful control options against unauthorized entry and change. Protection is provided by restricting access to control options.

### A.1.1 Restricted Options

Any control option that changes the security environment is protected. This includes all control options except those that only request CA-Top Secret status displays. These options are:

- FACILITY
- ST
- STAT
- STATUS
- VERSION

When a restricted control option is changed or specified at the O/S console, CA-Top Secret will display message TSS9080A as follows:

```
TSS9080A ENTER OPERATOR ID/PASSWORD FOR MODIFY ACCOUNTABILITY
```

This message prompts the user for one of the following authorizations before allowing the change to take effect.

## A.1.2 Authorization

At the prompt the person entering/changing a control option must either:

- Enter the MSCA's previous or current password, or
- Enter an ACID that possesses the CONSOLE attribute, followed by the ACID's password in the format ACID/password.

## A.1.3 CONSOLE Attribute

The CONSOLE attribute is attached to an ACID via a TSS ADDTO/REMOVE command function. CONSOLE is also used with the TSS CREATE command.

**Note:** To add the CONSOLE attribute to an ACID, the CA-Top Secret security administrator must have MISC9 administrative authority.

The CONSOLE attribute allows an operator to enter TSS control options at the console or via the TSS MODIFY command.

## A.1.4 Entry Methods

CA-Top Secret provides four methods that authorized operators can use to select and change control options:

- O/S MODIFY TSS commands
- O/S START TSS commands
- PARM field of TSS started task procedure
- Parameter File at startup of CA-Top Secret

The control options most often used by the operator are listed next. Remember that in some cases the operator must be granted the necessary authorization (the MSCA's previous password or through CONSOLE attributes to access these options).

- MODE
- BACKUP
- BYPASS
- CANCEL
- DUMP
- RECOVER
- REINIT
- SECTRACE
- SUSPEND
- SYNCH
- SYSOUT

No special authorization is required for an operator to use the following control options:

- STATUS
- VERSION

A brief description of each of the most often used control options appears next.

### A.1.5 Mode

CA-Top Secret supports four modes of operation for all facilities. The modes are DORMANT, WARN, IMPLEMENT, and FAIL. Modes are assigned at five different levels:

- GLOBAL**      The default for the entire CA-Top Secret community.
- FACILITY**    Affects a particular facility in the community.
- PROFILE**     Affects a particular group of users attached to the profile.
- USER**         Affects a particular user within the community.
- RESOURCE**   Forces a particular resource authorization to be processed in FAIL mode.

The global level is implemented via the MODE control option, or on a FACILITY level via the MODE= suboption of the FACILITY control option. The profile, user, and resource levels are implemented using the TSS PERMIT command. Refer to the *Control Options Guide* for details.

### A.1.6 BACKUP Control Option

The BACKUP control option allows the operator to:

- Request an immediate backup of the Security File.
- Select/change a time for an automatic daily backup.
- Deactivate the automatic backup.

To use the BACKUP control option, these procedures must be completed first:

1. Create a backup file which is the same size as the Security File during CA-Top Secret installation. Then copy the master Security File into the Backup Security File. See the section "TSSBCKUP Procedure" for details.

**Note:** CA-Top Secret will not back up the Security File on the same day that CA-Top Secret is started, unless it is started before any scheduled backup time. Refer to the *Installation and Maintenance Guide* for more details.

2. Enter the BACKUP DD statement into the CA-Top Secret started task procedure.

Specify the BACKUP control option as follows:

**BACKUP** To immediately backup the Security File.

**BACKUP(hhmm)** To select a time for CA-Top Secret automatic backup. (Default is 0100.)

**BACKUP(OFF)** To deactivate the CA-Top Secret automatic backup feature.

This control option can be specified through any of the entry methods listed at the beginning of this section.

**Examples:** To cause an automatic backup of the Security File to occur at 2 a.m. enter the following command:

```
TSS,BACKUP(0200)
```

To cause an immediate backup to occur enter:

```
TSS,BACKUP
```

## A.1.7 BYPASS Control Option

This control option permits the MSCA to request emergency security bypass for a specific job, all jobs, or a specific ACID. It is the responsibility of the security administrator to decide who is given access to this control option and which jobs or ACIDs should bypass security. Access to the BYPASS control option must be explicitly permitted.

When this option is in effect, an audit record is written, and messages appear on the security console indicating that security is being totally or selectively bypassed.

The BYPASS control option is entered via the MODIFY commands in one of the following formats as appropriate:

```
TSS,BYPASS(jobname or ACID)
```

```
TSS,BYPASS(RESET)
```

```
TSS,BYPASS(EVERYJOB)
```

| <b>Operand</b>  | <b>Description</b>                                                                                                                                                                                                                                                              |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>jobname</b>  | Enter the name of the batch job, started task procedure, or the ACID identifying the online session (or job), which will bypass security. Only one job or ACID can be specified per command entry; however, any number of ACIDs/jobs may be specified through multiple entries. |
| <b>RESET</b>    | Terminates security bypass for all jobnames and ACIDs that had been designated to bypass security.                                                                                                                                                                              |
| <b>EVERYJOB</b> | Allows ALL jobs and online sessions to bypass security. USE WITH EXTREME CAUTION.                                                                                                                                                                                               |

### A.1.8 CANCEL Control Option

The CANCEL control option allows use of the O/S CANCEL command to bring down the CA-Top Secret address space. The CA-Top Secret address space is eligible for cancellation only after specifying the CANCEL control option.

Specifying CANCEL has the immediate effect of deactivating the CA-Top Secret address space. The security interface is inactive and the DOWN options are **not** in effect. Refer to the *Control Options Guide* for information on DOWN option processing.

CANCEL should be used only in emergency situations it is not the recommended method for normal shutdown.

This option is entered via the TSS MODIFY command.

### A.1.9 DUMP Control Option

DUMP is used to produce a diagnostic dump of control blocks within the CA-Top Secret address space and common storage. This control option should only be used at the request of CA Technical Support.

Using this option requires a valid ACID with the CONSOLE attribute attached to it, and the correct response to the prompt for the ACID/password combination.

The DUMP control option is entered via a MODIFY command.

### A.1.10 RECOVER Control Option

This control option allows CA-Top Secret to record changes made to the Security File onto the Recovery File. Changes include those made automatically by CA-Top Secret (automatic volume ownerships, password changes) and those made by a CA-Top Secret security administrator via the TSS command.

| Operand    | Description                                                                                                                |
|------------|----------------------------------------------------------------------------------------------------------------------------|
| <b>ON</b>  | Activates the Recovery File. Indicates that the changes made to the Security File will be recorded onto the Recovery File. |
| <b>OFF</b> | Deactivates the Recovery File.                                                                                             |

The RECOVER control option is specified via all entry methods listed in the beginning of this section.

**Note:** If this control option is omitted at CA-Top Secret startup, RECOVER(ON) will be in effect if the CAIRECV DLBL statement is in the CA-Top Secret started task.

When running the TSSRECV utility, this control option must be set to RECOVER(OFF) to prevent double recording of changes. Refer to the *Installation Guide* for details.

### A.1.11 SECTRACE Control Option

The SECTRACE control option activates a diagnostic security trace on the activities of all defined users or of specific users. In most cases, the security administrator will delegate the responsibility of tracing users to the operations staff, so it is important for operators to be familiar with using the SECTRACE option. SECTRACE is usually used at the request of CA Technical Support.

SECTRACE is specified in the following format:

```
SECTRACE(WTO|WTL|OFF)
SECTRACE(ACT,WTO|WTL)
```

|            |                                                                                                                                                                                                                                                                                                                 |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>WTO</b> | Activates the trace and routes messages to the master console for all users and events.                                                                                                                                                                                                                         |
| <b>WTL</b> | Activates the trace and routes messages to the SYSLOG (system log).                                                                                                                                                                                                                                             |
| <b>OFF</b> | Deactivates diagnostic tracing.                                                                                                                                                                                                                                                                                 |
| <b>ACT</b> | Activates the trace for users that have the TRACE attribute attached to their ACIDs. The TRACE attribute is added via the TSS ADDTO command function. A CA-Top Secret security administrator or an operator must have MISC9 administrative authority to attach the TRACE attribute to ACIDs within their scope. |

SECTRACE can be specified through any of the entry methods listed at the beginning of this section.

**Note:** In TSO, trace information always goes to both the user's terminal and SYSLOG.

**Example:** To trace the jobs of USER01, the operator enters:

```
TSS ADD (USER01) TRACE
TSS MODIFY('SECTRACE(ACT,WTO)')
```

Refer to the *Troubleshooting Guide* for details on how to interpret the TRACE information.

### A.1.12 STATUS Control Option

This control option provides the current settings of various control options and information concerning cross-memory requests and Security File service requests. The STATUS control option can be entered as part of a TSS MODIFY command.

**Example:** To display system status, the operator enters:

```
TSS,STATUS
```

### A.1.13 SUSPEND Control Option

This control option is used by the operator to suspend any ACID. Suspension prevents the user from gaining access to the system through the suspended ACID. If a user is suspected of any subversive activity the operator can immediately suspend a suspicious user, without waiting for the MSCA to issue the control option.

Once suspended, the user must be cancelled off the system. The ACID must be unsuspended by the operator or security administrator through a TSS REMOVE command to reactivate the ACID. Note that the SUSPEND option cannot be used to unsuspend a user.

SUSPEND can be entered through a TSS command only.

**Example:** To SUSPEND USER01 enter:

```
TSS,SUSPEND(USER01)
```

### A.1.14 SYNCH Control Option

This option requests immediate synchronization of global in-memory tables (ALL, STC, AUDIT) with the Security File. SYNCH is only required for processors in DORMANT mode.

SYNCH is entered via a TSS command.

**Example:** To synchronize global in-memory tables the operator enters:

```
TSS,SYNCH
```



### A.1.15 SYSOUT Control Option

Specifying the SYSOUT control option causes the Activity Log (which records console communications and unexpected events) to be spun off. A new log is dynamically allocated.

SYSOUT is entered via an TSS MODIFY command.

**Example:** To record console communications, the operator enters:

```
TSS,SYSOUT
```

### A.1.16 VERSION Control Option

This option displays CA-Top Secret's version. VERSION is entered via a TSS MODIFY command.

**Example:** To display the current version, the operator enters:

```
TSS,VERSION
```

The following message is displayed:

```
TSS9500I VERSION=3.0.000 yymmK0510 LAST MAINT=mm/dd/yy, hh.mm.ss
```

### A.1.17 TSS MODIFY Command

The TSS MODIFY command is used to enter or change a control option. This command is entered from the console and requires an access authorization. If an operator attempts to alter a restricted control option, CA-Top Secret will prompt for an authorized ACID/password combination before processing the change.

TSS MODIFY commands may be entered by using the following notation:

```
TSS MODIFY option(operand if required)
```

**Example:** To indicate tape security is in effect enter:

|

TSS MODIFY TAPE(DSN)

## A.2 Startup Procedures

CA-Top Secret should start up automatically during the IPL process after CA-CIS initialization (this should be the case no matter which IPL parameters are used to bring up your system) or it can be started as a subsystem. In any event, CA-Top Secret should be the first started task to execute during a system IPL and the last address space to be brought down at the end of the day.

During startup, CA-Top Secret may issue VSE commands after security has been initialized into VSE. This ensures that security is always present after POWER initialization.

## A.3 Shutdown Procedures

CA-Top Secret provides two types of shutdown; normal or temporary and end-of-day. This section will describe shutdown procedures and their use.

### A.3.1 Temporary Shutdown

A temporary or normal shutdown is the recommended method of deactivating the CA-Top Secret address space.

To temporarily shutdown the CA-Top Secret address space, the operator enters:

```
TSS SHUTDOWN
```

CA-Top Secret displays the following CA-Top Secret messages:

```
TSS9072I ** SELECT TYPE OF SHUTDOWN ** <I> TO IGNORE
TSS9072I <Z> END OF DAY; RE-IPL WILL BE REQUIRED
TSS9072A <T> TEMPORARY; MAY IMPACT THROUGHPUT
```

For a normal or temporary shutdown, the operator enters:

```
T
```

### A.3.2 End-of-Day Shutdown

**An end-of-day shutdown deactivates the CA-Top Secret address space and requires an IPL.** End-of-day shutdown prohibits new initiations in all modes other than DORMANT. This means new users will not be able to sign on to any facility and new batch jobs will not execute.

To shutdown the CA-Top Secret address space, the operator enters:

```
TSS SHUTDOWN
```

CA-Top Secret displays the following CA-Top Secret messages:

```
TSS9072I ** SELECT TYPE OF SHUTDOWN ** <I> TO IGNORE
TSS9072I <Z> END OF DAY; RE-IPL WILL BE REQUIRED
TSS9072A <T> TEMPORARY; MAY IMPACT THROUGHPUT
```

For an end-of-day shutdown, the operator enters:

```
Z
```

Provided the operator has the authority, an end-of-day shutdown can be reset using the RESETEOD control option. This control option allows CA-Top Secret to be restarted, without IPLing, after it has been brought down (accidentally) for end-of-day shutdown. See the section, "Restarting CA-Top Secret After End-of-Day" for procedures.

## A.4 Restart Procedures

There are four ways to restart CA-Top Secret after it has been brought down:

1. Restarting CA-Top Secret after IPL.
2. Restarting CA-Top Secret after temporary shutdown.
3. Reinitializing CA-Top Secret after temporary shutdown.
4. Restarting CA-Top Secret after end-of-day shutdown.

The remainder of this section discusses these four methods and the conditions in which they are most likely to be used.

## A.5 Security File Backup Procedures

This section describes backup procedures for the Security File. There are two backup procedures provided by CA-Top Secret:

1. Automatic backup to DASD
2. Tape backup

The Security File is the database containing all security-related information about users, profiles, departments, divisions, and resources. The CA-Top Secret utility, TSSMAINT, creates and formats the Security File. Usually, operations is not responsible for these procedures; however, familiarity with the Security File format may prove helpful. Refer to the CA-Top Secret *Installation Guide* for details.

Because the Security File is a critical resource, it should be integrated into your standard backup procedures. The loss of the Security File database would necessitate running your system without security until the Security File is rebuilt or recovered. This would result in countless security exposures.

### A.5.1 CA-Top Secret Automatic Backup Feature

CA-Top Secret provides an automatic backup feature used to dynamically back up the Security File to DASD.

A daily backup of the Security File via the CA-Top Secret automatic backup feature with the TSS RECOVERY procedure, guarantees no loss of the file; see the section "Recovery Procedures" for details.

To implement the use of the CA-Top Secret automatic backup feature, the CA-Top Secret security administrator or systems programmer must create the Backup Security File on an alternate DASD volume prior to backup. This should be done during installation.

It is also recommended that this Backup Security File exist on a different string with a different control unit than the Security File. This will ensure that the Backup Security File will be available in case of a hardware failure. The Backup Security File is a copy of the Security File, and as such it must be considered a sensitive, high risk data set. The Recovery File should be located on the same pack as the Backup File. The *Installation Guide* contains the procedure for creating the Backup Security File.

Using the CA-Top Secret automatic backup feature is the recommended method for backing up to DASD. This feature requires only:

- The control options RECOVER(ON) and BACKUP(hhmm) specified.

For information about setting these control options see the "BACKUP" and "RECOVERY" control options at the beginning of this appendix.

Note that a backup can be performed at any time by using:

|                   |
|-------------------|
| TSS MODIFY BACKUP |
|-------------------|



### A.5.1.1 TSSBCKUP Procedure

TSSBCKUP can be used to back up the:

- Security File (optional if CA-Top Secret automatic backup is in effect)
- Recovery File
- Audit/Tracking File

**Note:** The Audit/Tracking File can be backed up in its entirety or it can be archived daily, weekly, monthly, etc., using the TSSARCHI JCL found in the SAMPJCL library.

The TSSBCKUP JCL allows the operator to manually backup a DASD file to tape. It uses the IBM utility FCOPY to copy the files.

Sample JCL for TSSBCKUP is found in the *Installation Guide*. The CA-Top Secret security administrator or systems programmer edits the JCL at installation.

TSSBCKUP is run as a batch job. For backing up files at regular intervals, you should run TSSBCKUP as a batch job through a Job Scheduler.

### A.5.1.2 Manual Backup of the Security File

Even with the CA-Top Secret automatic backup feature in effect, it may be necessary for your operator to use TSSBCKUP to copy the Backup Security File to tape. If your site has only one copy of the Backup Security File, the operator should run TSSBCKUP prior to using the TSSRECV utility. This prevents loss of data in the case of hardware malfunction and provides a reserve copy of the Backup Security File on tape.

TSSBCKUP should be set up to back up the Security File by default. If it is necessary to back up any of the other CA-Top Secret files, simply specify the correct file when executing the task.

## A.6 Recovery and Audit/Tracking File Backup Procedures

### A.6.1 Recovery File

The Recovery File records changes made to the Security File. It is a wraparound file; when the file is full, recording continues at the beginning of the file, overlaying existing data. The Recovery File is created and formatted using the TSSMAINT utility; refer to the *Installation Guide* for details.

Once the Recovery File is created, the recorded changes are written to the Security File by running the CA-Top Secret utility TSSRECVR. The TSSRECVR utility and its use is detailed in the *Installation Guide*.

It is recommended that the Recovery File be allocated on a different volume than the Security File to ensure that if a hardware malfunction occurs recovery using an alternative device or path is possible. The size of the Recovery File is dependent on the interval between Security File backups. The file should be made large enough to record two or three days worth of changes for every day in the security backup interval. For example, if the Security File is backed up at the end of each day, the Recovery File should be large enough to accommodate at least two days worth of changes. Refer to the section, "BACKUP Control Option" for details on backup intervals.

There is no need to backup the Recovery File.

### A.6.2 Audit/Tracking File(s)

The Audit/Tracking File(s) are used to record security incidents. Violations and audited events are held in this wraparound file. Incidents can be generated by report or displayed online as the events occur.

It is not necessary to backup the Audit/Tracking File(s).

## A.7 Security File Serialization

When a system is attempting to acquire the Security File lock, and detects that the lock is held, CA-Top Secret retries to acquire the lock for several minutes. If the lock is still not obtained, CA-Top Secret issues the TSS9123 message which prompts for a reply of WAIT or RESET. The correct reply to the TSS9123 message is WAIT. This indicates that CA-Top Secret will wait for the holder to release the lock.

The only situations in which replying RESET would be appropriate are:

1. Having multiple systems sharing the Security File and all of them having the TSS9123 prompt outstanding. The correct action is to respond RESET to the first system, and WAIT on all remaining systems.
2. If the system identified in the TSS9123 message took a hardware failure and is non-operational.

Replying RESET any other time directs one system to ignore whatever the other system is doing to the Security File. As a result, this introduces the potential of damaging your Security File.

## A.8 Messages and Codes

For a complete listing of CA-Top Secret messages and codes, the security administrator should direct the operator to the *Messages and Codes Guide*. This guide defines all abend codes, error codes, and information messages that may be encountered during the use of CA-Top Secret.

## Appendix B. Prefixed Resources

---

CA-Top Secret manages two distinct types of resources: general and prefixed. Prefixed resources are those that support masking characters, and are listed below.

APPCID  
APPCLU  
APPCSI  
APPCTP  
DB2BASE  
DB2COLL  
DB2PKG  
DLFCLASS  
DSNAME  
DTADMIN  
DTSYSTEM  
DTTABLE  
DTUTIL  
JESJOBS  
JESSPOOL  
MQCONN  
MQNLIST  
MQPROC  
MQQUEUE  
NODES  
OPERCMD5  
PANAPT  
SDSF  
VMMDISK  
VSESLIB  
VSEMEMBR  
WRITER

Prefixed resources have an attribute of MASK when the RESCLASS is displayed using the TSS LIST(RDT) RESCLASS(resource class) command.



# Glossary

---

CA-Top Secret Glossary

**abend.** Abnormal ending. An early termination of a program due to an error.

**access.** An ACID's ability to use a resource.

**ACID.** A unique character-string identifier by which CA-Top Secret identifies a user's Security Record.

**ACID Authorities.** Specifies what actions a security administrator can perform on ACIDs within his scope.

**ACID Type.** Determines an ACID's function in the Security File structure. Types include: User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA.

**ACTION.** A CA-Top Secret command keyword used to refine a user's ability to access resources.

**Administrative Authority.** The different classes of authority, assigned via the TSS ADMIN command function, used to determine what functions a security administrator can perform.

**ALL Record.** Used to record global access requirements that are effective for all users, both defined and undefined to CA-Top Secret.

**attribute.** A specific authority, privilege, or restriction that is assigned to an ACID.

**Audit/Tracking File.** Contains information on resource access events. It can provide both online and batch reports.

**authorization.** CA-Top Secret's way of allowing access to a protected resource.

**batch.** A method of processing large amounts of data at one time for jobs too large to perform immediately online.

**BDAM.** Basic Direct Access Method. Used to directly retrieve or update blocks of information stored on a direct access device.

**CA-Roscoe.** A time-sharing software system (similar to TSO), enabling two or more users to execute their programs at the same time.

**Centralized Administration.** With centralized administration, only central level administrators make or implement security decisions.

**CICS.** Customer Information Control System. A teleprocessing monitor used for a variety of applications.

**control block.** A storage area that a program uses to hold control information.

**control options.** A mechanism whereby CA-Top Secret can customize your security environment.

**CPU.** Central Processing Unit. The "brain" of the computer, consisting of three basic parts: the control unit, the arithmetic/logic unit, and the storage (main memory) unit. The control unit interprets instructions and issues the appropriate commands to the other two parts as well as to other computer system devices. The storage unit holds instructions and data required for use in the system temporarily and acts as the common link to all parts of the system. The arithmetic/logic unit performs all the necessary arithmetical and logical manipulations on data supplied to the system.

**DATA authorities.** Determines which portion of a Security Record a TSS administrator can display.

**database.** A systemized collection of data stored for immediate access.

**data set.** A group of logically related records stored together and given a unique name.

**DCA (Department Control ACID).** Used where security administration has been decentralized. The DCA's scope of authority is limited to his assigned department.

**Decentralized Administration.** With decentralized administration, security decisions are handled by security administrators at all levels of the CA-Top Secret security structure.

**default.** A value or action automatically supplied by the computer system unless you specify some other alternative.

**department.** A mandatory collection of users and profiles defined by a department ACID. It cannot sign on, and does not have a password, but can own resources.

**division.** An optional collection of departments defined by a divisional ACID. It cannot sign on, and does not have a password, but can own resources.

**DORMANT mode.** CA-Top Secret is installed, but is not actively validating access.

**Dynamic Update Facility (DUF).** An attribute that allows for the extraction and updating of installation-selected data for any designated users.

**encryption key.** An installation-defined hexadecimal string used to encrypt all information in the Security File.

**EXpire.** A keyword used when creating a password for the first time, or when replacing a password. It causes the created password to expire the first time it is used.

**facility.** CA-Top Secret considers a facility as a way of grouping options and associating them with a particular service that users sign on to. Examples of such services are BATCH, Started Tasks, CICS, and TSO.

To sign on to a service, a user must be given access to the facility. Some services—such as BATCH, STC, and TSO—are automatically associated with the facility of the same name. Others—such as CICS, CA-IDMS, and IMS—must be associated with an appropriate facility.

**Facility Matrix Table.** A table containing all facilities defined to CA-Top Secret. Each entry contains information about the specific attributes associated with a particular facility (i.e., VM, TSO, CICS/PROD) and can be viewed and modified via the FACILITY control option.

**FAIL mode.** CA-Top Secret is in full control of all access requests. Violations result in termination of the request.

**FDT (Field Description Table).** A table containing all pre- and dynamically-defined fields identified to CA-Top Secret.

**generic prefix.** A shorthand way to specify a number of similarly-named resources in the same class.

**global access.** Any access specified in the ALL Record.

**group.** A collection of User ACIDs who can share access authorities for protected resources. Groups are used in the OpenMVS environment only.

**IMPLEMENT mode.** CA-Top Secret is actively validating requests for resources and facilities, and will fail unauthorized requests by ACIDs that have been defined.

**inactive.** Defines the number of days before CA-Top Secret will deny use of an ACID with an expired password.

**interval.** The period of time after which a password must be changed. Passwords not changed by the end of this period of time will expire.

**JCL.** Job Control Language. The computer language that links your program to the computer, assigning files to specific devices and describing each file in detail to the system.

**Limited Command Facility (LCF).** A facility that allows the security administrator to control use of commands/transactions (TSO, CA-Roscoe, CICS, etc.) available to a user.

**locktime.** The period of time after which a terminal automatically locks if no transactions or commands are issued. The user must issue TSS UNLOCK and supply a valid password to unlock the terminal.

**LSCA (Limited Central Security Control ACID).** This control ACID can have all the authority of an SCA, but unlike the SCA, the LSCA can have a limited scope of control. Only the MSCA can determine that scope and it can encompass ZCAs, VCAs, DCAs, Profiles, Users, and other LSCAs.

**masking.** A technique that can be used to reduce the number of definitions which need to be made to CA-Top Secret.

**MISC1.** Specifies what miscellaneous authorities a security administrator has. For example, LCF, INSTDATA, RDT, and TSSSIM to name a few.

**MISC2.** Specifies what authorities a security administrator has for SMS-, VAX-, and TSO-related resources.

**MISC8.** Specifies what authorities a security administrator has regarding PassTickets.

**MISC9.** Specifies what miscellaneous authorities a security administrator has. For example, BYPASS, TRACE, CONSOLE, and STC to name a few.

**MGMTCLAS.** A pre-defined resource in the RDT that controls user access to a management class defined by the storage administrator. Each unique management class is identified by a unique one- to eight-byte name.

**mode.** Specifies the level of security under which CA-Top Secret is operating.

**MSCA (Master Security Control ACID).** The one Control ACID that is predefined, active, and assigned full administrative authority the first time CA-Top Secret is started. This administrator's scope of authority includes the entire installation. The MSCA can designate and authorize SCAs, LSCAs, ZCAs, VCAs, and DCAs.



**MVS.** Multiple Virtual Storage. A virtual storage operating system that IBM supports.

**MVS.** Multiple Virtual Storage. A virtual storage operating system that IBM supports.

**NDT (Node Description Table).** A table containing all PassTicket information.

**online.** A method of immediately processing information at the time of data entry. Usually, the results are displayed on the screen.

**OS modify.** In an MVS or VS1 system, use this command to communicate with OMS/Monitor.

**ownership.** When a resource is owned by a particular ACID, that ACID has unlimited access to the resource. Ownership defines the resource to CA-Top Secret. All other ACIDs must be specifically authorized to access the resource.

**Parameter File.** Standard CMS file containing all CA-Top Secret control options and used to establish the operating environment of CA-Top Secret at startup.

**password masking.** Allows the user to specify the type of character that is accepted for each position in the password.

**permissions.** The process that makes an owned resource available to other users in a controlled manner.

**prefixing.** Allows a group of similar resources of the same type to be defined to CA-Top Secret simultaneously.

**profile.** An ACID containing a collection of access characteristics common to several users. It generally describes the access characteristics of a particular job function. A profile cannot sign on, and does not have a password. Up to 254 profiles can be attached to a user's ACID. Any number of users can be associated with a single profile.

**program pathing.** This access authorization option lets you permit access to a data set only through a specific program using the PRIVPGM parameter on a PERMIT.

**random password generation.** An option that allows you to specify that CA-Top Secret will supply each user with new randomly generated passwords.

**RACF (Resource Access Control Facility).** An IBM program product that provides system entry, resource access control, auditability, accountability, and administrative control.

**RDT (Resource Descriptor Table).** A table containing all pre- and dynamically-defined resources identified to CA-Top Secret.

**Record.** There are several different record types in CA-Top Secret. They include: the main Security Record for each ACID, the Audit Record, the ALL Record, the Profile Record, the Department, Division, and Zone Records, the Resource Descriptor Table Record, and the Control ACID's Records.

**Recovery File.** Contains a record of all changes made to the Security File. It can be used to recreate the Security File if it becomes damaged or unusable because of hardware or software problems.

**RESOURCE authorities.** Indicates the kind of resource maintenance authorities that are to be given to a particular security administrator.

**restricted password list.** A list of password prefixes that CA-Top Secret will not let users enter as passwords.

**SCA (Central Security Control ACID).** This type of administrator's scope of authority includes the entire installation. An SCA can designate and authorize VCAs and DCAs.

**Scope of authority.** Defined as what logical units the user has administrative control over.

**security administrator.** A person who is primarily responsible for such implementation and maintenance functions as defining users, resources, and levels of access. His administrative authority determines what he can do.

**Security File.** An encrypted Security Database consisting of the Security Records which contain all user and resource permissions and restrictions.

**Security Record.** A part of the Security File that contains a set of user and profile records copied into a user's address space, including such information as resources a particular user can access and how he can use them. Also contains user characteristics, authorities, and so on.

**Security Validation Algorithm.** Determines whether CA-Top Secret should accept or deny users' requests to use a resource (such as CP commands, minidisks, data sets).

**SMF.** An optional control program feature providing gathering and recording information that can be used to evaluate system usage.

**source of origin.** The source (a terminal or reader) from which an access request is made.

**Super ACID.** Used to solve off-hours production problems, it is an ACID which the operator (if authorized) can assign to an abending job to allow it to run. It may also be used for sign on by a programmer who has been temporarily assigned to solve problems.

**terminal locking.** Prevents unauthorized use of a terminal which is signed on and unattended. It can be set either automatically or manually.

**SMSDATA.** The default DATA CLASS keyword that has been added to an ACID. The value of this field is extracted by CA-Top Secret from the RESOWNER ACID of the data set in allocation, and is returned as the &DEF\_DATACLAS value to the ACS routine.

**SMSMGMT.** The default MANAGEMENT CLASS keyword that has been added to an ACID. The value of this field is extracted by CA-Top Secret from the RESOWNER ACID of the data set in allocation, and is returned as &DEF\_MGMTCLAS value to the ACS routine.

**SMSAPPL.** The default APPLICATION ID keyword that has been added to an ACID. The value of this field is extracted by CA-Top Secret from the RESOWNER ACID of the data set in allocation, and is returned as the &APPLIC value to the ACS routine.

**SMSSTOR.** The default storage keyword that has been added to an ACID. The value of this field is extracted by CA-Top Secret from the RESOWNER ACID of the data set in allocation, and is returned as the &DEF\_STORCLAS value to the ACS routine.

**STC.** A pre-defined entry in the Facilities Matrix Table that defines a started task.

**STORCLAS.** A pre-defined resource in the RDT that contains user access to a specific storage class defined by the storage administrator. Each unique storage class is identified by a unique one- to eight-byte name.

**TSO.** Time sharing option. Enables two or more users to execute their programs at the same time by dividing the machine resources among terminal users.

**TSS ADD.** A command function that adds ownership of resources and special authorizations to an ACID.

**TSS ADMIN.** A command function that grants administrative authority.

**TSS Command.** The means by which the security administrator communicates with CA-Top Secret.

**TSS Command Functions.** When combined with the TSS command, they specify what the administrator chooses to perform.

**TSS CREATE.** A command function that creates an ACID.

**TSS DEADMIN.** A command function that removes administrative authority.

**TSS DELETE.** A command function that deletes an ACID.

**TSS HELP.** A command function that displays a description of TSS command parameters.

**TSS LIST.** A command function that displays the Security Record of a specific ACID; the Security Record of all ACIDs in a department, division, or profile; the AUDIT, ALL, RDT, or STC Records.

**TSS LOCK.** A command function that locks a terminal and prevents its use until a valid password is entered in response to the TSS UNLOCK function.

**TSS MODIFY.** A command function that displays or changes CA-Top Secret's system status and options.

**TSS MOVE.** A command function that moves an ACID from one department, division, or zone to another.

**TSS PERMIT.** A command function that authorizes resource access.

**TSS REMOVE.** A command function that removes resource ownership and special authorizations from an ACID.

**TSS RENAME.** A command function that changes the name of an ACID.

**TSS REPLACE.** A command function that changes information in an ACID.

**TSS REVOKE.** A command function that removes resource access authorization.

**TSS UNLOCK.** A command function that unlocks a terminal.

**TSS WHOAMI.** A command function that displays current user information.

**TSS WHOHAS.** A command function that displays who has access to a specific resource.

**TSS WHOOWNS.** A command function that displays resource ownership information.

**TSSAUDIT.** A batch utility that allows auditors to report on changes made to the Security File and sensitive MVS facilities.

**TSSCFILE.** A batch utility that generates a fixed format output file whose records closely parallel the output of a TSS LIST command function.

**TSSCHART.** A batch utility that provides an overview of the Security File architecture through the generation of block charts of ACIDs/owned resource relationships.

**TSSCVUAD.** A batch utility that creates TSS commands to add all of the users on UADS to the CA-Top Secret Security File.

**TSSPROT.** A utility to determine which data sets have (and do not have) their security bit indicators turned on in an SU32 environment.

**TSSREPORT.** A batch utility that applies the capabilities of CA-Earl, an easy-to-use report language, to the output of the TSSCFILE utility providing formatted summaries of CA-Top Secret data.

**TSSRCNVT.** A utility that constructs TSS commands to create divisions, departments, users, profiles, and authorizations for data sets from IBM's Resource Access Control Facility (RACF) data sets, and optionally, from user-written control statements.

**TSSSIM.** A utility that provides the ability to test permissions on the Security File without affecting the "real" production environment. In addition, it can also be used as a diagnostic tool when there is a need to debug errors in the Security File permissions.

**TSSTRACK.** A utility used to monitor security-related events from a TSO or CICS terminal in realtime.

**TSSUTIL.** A report generator/extract utility that can be used to produce batch reports of security-related events that have been logged to the Audit/Tracking File or SMF.

**TZONE.** A CA-Top Secret option that compensates for security environments which cross time zones.

**UADS (User Attribute Data Set).** In TSO, a partitioned data set with a member for each authorized user. Each member contains the appropriate passwords, user definitions, account numbers, LOGON procedure names, and user characteristics that define the user profile.

**User.** The lowest ACID level in the security structure. Generally, a User can sign on, and initiate jobs, has a password, and belongs to a department.

**user exits.** Computer instructions that allow you to modify the software package's code to meet your data center's needs.

**VCA (Divisional Control ACID).** This ACID is used where security administration has been decentralized. The VCA's scope of authority is limited to his assigned division, including the departments attached to it.

**violation.** An unauthorized attempt to access a protected resource.

**VSAM.** Virtual Storage Access Method. Allows the transfer of information between the CPU's main memory and a direct access storage device. Records organized in logical key field sequence, in physical creation sequence, or by relative-record number, can be accessed directly or sequentially.

**WARN mode.** The mode in which CA-Top Secret is validating access, but violations do not result in a request being failed.

**XAUTH.** Allows the designated security administrator to permit or revoke access to an owned resource.

**ZONE.** An optional collection of divisions defined by a zone ACID. It cannot sign on, and does not have a password, but it can own resources.

**ZCA (Zone Control ACID).** An administrative ACID whose scope of authority includes an entire zone.



# Index

---

## Special Characters

\$\$LOG\$\$ file 1-21

## A

ABSTRACT keyword 6-18, 7-32

ABSTRACT resource 6-22

### Access

authorization 6-21

designing 6-21

multiple 6-27

revoking multiple 6-28

special cases 7-39

controlling 6-21

denying 6-34

how to authorize 6-24

level, keywords 6-30

removing from former owner 6-16

restricting

by facility 6-29

by level 6-30

by path 6-30

options for 6-28

with ACTION keyword 6-33

with PERMIT keywords 6-21

ACCESS keyword 6-30

### Access level

backing up volume 7-6

combining user defined and standard 6-33

data set 7-13

default 7-13

defining unique for dynamically defined resource 6-32

FETCH-only 6-31

FIELD 7-34

placement of multiple 6-32

RACHECK comparison 6-32

record and field level control 6-31

restoring volume 7-6

restricting 6-31

transactions 7-38

URI/UR2 7-34

volume 7-5

ACCESS(NONE) 6-34

### ACEE

definition of 11-39

Master ACEE 11-39

multiple user address space 11-38

ACEE (*continued*)

use of 11-39

USER C/B 11-40

ACEE= parameter requirements 11-23

ACEP facility defaults 10-9

### ACID

authorities 4-9

automatic inactivity suspension 4-27

creating model 4-23

creating TYPE(DCA) 4-3

creating TYPE(DEPT) 4-19

creating TYPE(DIV) 4-19

creating TYPE(GROUP) 4-33

creating TYPE(LSCA) 4-3

creating TYPE(PROFILE) 4-29

creating TYPE(SCA) 4-3

creating TYPE(VCA) 4-3

creating TYPE(ZCA) 4-3

creating TYPE(ZONE) 4-19

creation without password 5-5

defined 1-6

defining 4-19, 4-29

defining group 4-33

deleting 4-25

demoting 4-24

Department 1-7

derivation 10-24

Division 1-7

expiration 4-26

Group 1-7, 4-33

hierarchy 1-10

limited duration suspensions 4-27

maintaining 4-9

maximum number 4-29

password history 5-8

Profile 1-7, 4-29

promoting 4-24

reinstating suspended 5-15

required administrative authority 4-25, 4-27

substitution 6-12

types, defining 1-7

User 1-7

validating 1-13

Zone 1-8

ACID keyword, operands 4-9

ACID resource 6-22

- ACTION exit point 11-57
- ACTION keyword
  - ADMIN operand 6-34
  - AUDIT operand 6-34
  - DENY operand 6-34
  - EXIT operand 6-34
  - FAIL operand 6-34
  - NODSN operand 6-34
  - NOTIFY operand 6-34
  - PASSWORD operand 6-34
  - using 6-33
  - VMPRIV operand 6-34
- ACTION parameter 2-4
- ACTION(NODSN) 7-8
- Activating TSS A-13
- Adding temporary groups 4-34
- Adding temporary profiles 4-30
- Administrative authorities 1-11
- Administrative authority
  - ACID keyword 4-9
  - DATA keyword 4-10
  - FACILITY keyword 4-13
  - MISC1 keyword 4-14
  - MISC2 keyword 4-15
  - MISC3 keyword 4-16
  - MISC8 keyword 4-16
  - MISC9 keyword 4-17
  - RESOURCE keyword 4-11
  - SCOPE keyword 4-18
- ALL Record 1-12, 1-22, 6-26
  - adding resource to 6-25
  - overriding access 6-26
- ALLMERGE 6-38
- ALLMERGE attribute 8-4
- ALLOVER 6-38
- APPC facility defaults 10-9
- APPCLU record 1-22
  - specifying CONVSEC 8-41
  - specifying INTERVAL 8-41
  - specifying SESSKEY 8-41
- APPCPORT resource 6-22
- APPCSI resource 6-22
- APPCTP resource 6-22
- APPL keyword 6-17, 7-29
- Applicable keywords
  - used with the FDT Record 8-13
- Application
  - authorizing access 7-30, 7-34
  - generic prefixing 7-29
  - removing ownership 7-29

- Application interface
  - Assembler example 11-17
  - COBOL example 11-19
  - code logic 11-3
  - command checking 11-11
  - data set checking 11-10
  - DUF (Dynamic Update Facility) 11-13
  - DUFUPD 11-14
  - DUFXTR 11-13
  - extracting field data 11-15
  - facility checking 11-11
  - facility list check 11-11
  - Field Extract 11-15
  - FLDUPD 11-15
  - FLDXTR 11-15
  - how to use 11-3
  - INSTDATA 11-13
  - invoking 11-2, 11-3
  - list of facilities 11-11
  - list of resources 11-12
  - log user information 11-13
  - other ACID checks 11-16
  - parameter list 11-3
  - performing checks 11-8
  - PL/I example 11-18
  - request fields 11-4
  - request record 11-3
  - request record field characteristics 11-4
  - required components 11-3
  - resource checking 11-9
  - resource list check 11-12
  - retrieve ACID 11-16
  - return codes 11-7
  - return fields 11-5, 11-6
  - sample code 11-3
  - status return codes 11-8
  - transaction checking 11-11
  - TSSAI 11-3
  - updating field data 11-15
  - updating installation data 11-14
  - uses of 11-2
  - volume checking 11-10
- Application protection 7-29
- APPLICATION resource 6-22
- AREA keyword 6-18
- AREA resource 6-22
- Assigning, ownership 6-4
- ATS 5-23
- Attaching groups 4-34
- Attaching profiles 4-30

ATTR 8-7, 8-14  
ATTR keyword 6-40  
ATTR, operands 8-4  
Attributes 7-2, 7-3  
Audit record 1-22  
Audit requirements 2-30  
Audit/Tracking file 1-20, 2-27, A-20  
Auditors 4-7  
Auditors, defining 2-30  
AUTH control option 6-37  
    processing considerations 6-39  
    setting by resource class 6-40  
Authentication devices, special 5-26  
Authority levels  
    MISC1 A-10  
    MISC9 A-8  
Authorizing resource access 6-24  
Automatic Terminal Signon (ATS) 5-23  
AVPV 8-41

## B

BACKUP control option A-5, A-18  
Backup file 1-21  
Batch  
    ACID derivation 10-24  
    ACIDs 2-15  
    authorization restrictions 10-23  
    card and remote reader 10-24  
    DEFACID control option 2-15  
    job scheduling 10-25  
    job submission 10-25  
    password 10-24  
    passwords 2-16  
    production scheduling 2-14  
    protection of 2-15  
    signon 10-23  
BATCH facility defaults 10-9  
Batch utilities  
    for logging 3-4  
    TSSAUDIT 3-4  
    TSSCPR 3-4  
    TSSRECV 3-5  
Best fit 6-1  
    criteria used 6-46  
    determining 6-46  
BLP for OS/390 2-22  
BLP, access level 6-31  
Bypass  
    attributes 7-2  
    resource checking 7-2

BYPASS control option A-6  
Bypass Label Processing (BLP) 7-19  
Bypassing security checking 7-3, 7-8, 7-17

## C

CA-CIS Architecture 1-3  
CA-IDMS  
    administration 10-22  
    area 10-22  
    authorization restrictions 10-21  
    password 10-21  
    program 10-22  
    signon 10-21  
    subschemata 10-22  
    terminal 10-22  
    transaction 10-21  
CA-Top Secret  
    controlling system entry 5-17  
    description of 1-5  
    hierarchy 1-10  
CA7 facility defaults 10-9  
CACMD resource 6-22  
CALENDAR records 8-20, 8-28  
CANCEL control option A-7  
CAPCID resource 6-22  
CAPCLU resource 6-22  
CHANGE exit point 11-56  
Changing an expiration date 4-27  
CICS 2-17  
    administration 10-18  
    application interface 10-18  
    authorization restriction 10-18  
    job submission 10-18  
    multiple signons 10-19  
    multiple user address space 10-1  
    password 10-18  
    program 10-18  
    security key 10-19  
    signon 10-17  
    terminal 10-18  
CICSPROD facility defaults 10-10  
CICSTEST facility defaults 10-11  
CMD keyword 6-18, 7-39  
COMMAND exit point 11-51  
Command Propagation Facility  
    *See* CPF  
Command Propagation Facility, *see* CPF 1-4  
Command Propagation Files 1-21  
COMPLETE facility defaults 10-11  
CONSOLE attribute 4-44, A-4, A-7, A-11

- CONSOLE facility defaults 10-11
- CONTROL access level 6-31
- Control ACID
  - administrative authority 1-8
  - definition of 1-8
- Control ACIDs 1-11
- Control ACIDs, creating 4-3
- Control options
  - AUTH 6-37
  - authorization A-4
  - default values and functions 4-43
  - FACILITY, operands 10-3, 10-6
  - NEWPW 5-16
  - restricted A-3
  - syntax xx
  - VMFAC 10-27
- CONVSEC keyword (MVS)
  - ADD function 8-41
  - REMOVE function 8-41
- CPCMD keyword 6-20
- CPCMD resource 6-22
- CPF 1-4
  - architecture 9-3
  - keywords used with 9-7
  - overview 9-2
  - recovery file 9-12
- CPF exit point 11-59
- CPU
  - access restrictions 6-13
  - defining 6-13
  - resource 6-22
- CPU keyword 6-17, 6-20
- CPUs, security for 2-23
- Creating
  - Control ACIDs 4-3
- Creating Control ACIDs 4-3
- Creating model ACID 4-23

## D

- DATA keyword
  - authorities 4-10
  - operands 4-10
- Data set
  - access levels 7-13
    - for creating 7-15
    - for VSAM 7-14
  - accessing all 7-16
  - assigning ownership 6-14
  - authorizing access 7-13
  - bypassing checking from volume 7-17
  - bypassing security 7-17
- Data set (*continued*)
  - bypassing security checking 7-17
  - creation authorization 7-15
  - default protection 7-10
  - generic prefix 7-11
  - masking 6-9, 7-12
  - ownership of 7-10
  - program pathing 7-15
  - removing ownership 7-10
  - securing relations to volume security 6-42
  - volume, versus 7-20
  - VSAM 7-14
- Data set protection
  - extending default 7-10
  - how to 7-10
- Data set/Volume access authorization relationships 6-42
- DATABASE resource 6-22
- DATASET exit point 11-50
- DB2 resource 6-22
- DB2BUFFP 6-19
- DB2BUFFP resource 6-22
- DB2COLL 6-19
- DB2COLL resource 6-22
- DB2DBASE 6-19
- DB2DBASE resource 6-22
- DB2PKG 6-19
- DB2PLAN 6-19
- DB2PLAN resource 6-22
- DB2PROD facility defaults 10-12
- DB2STOGP 6-19
- DB2STOGP resource 6-22
- DB2SYS 6-19
- DB2SYS resource 6-23
- DB2TABLE 6-19
- DB2TABLE resource 6-23
- DB2TABSP 6-19
- DB2TABSP resource 6-23
- DB2TEST facility defaults 10-12
- DBD keyword 6-17
- DBD resource 6-22
- DCA
  - See* Departmental Control ACID
- DCSS keyword 6-20
- DCSS resource 6-23
- DCT keyword 6-17
- DCT resource 6-23
- DEFACID 5-23
- Default access level 7-13
- Default protection 2-11
  - for resources 6-22
  - how to provide 6-22
  - for volumes 7-5



- Defining readers 7-23
- DEFNODES keyword 9-7
- DEFPROT 2-11
- DEFPROT attribute 6-2, 6-22, 7-5, 7-10, 7-26, 8-4
- Deleting an ACID 4-25
- Demoting an ACID 4-24
- Denying resource access 6-34
- Department
  - definition of 1-7
- Departmental Control ACID (DCA) 4-7
- DEVICES resource 6-23
- DIAG keyword 6-20
- DIAGNOSE resource 6-23
- DIRECTRY resource 6-23
- Disaster recovery 2-10
- DISPLAY 8-13, 8-16
- Displaying
  - ACID information 4-41
  - ACID Security Record information 4-37
  - ALL Record 4-37
  - Audit Record 4-37
  - DLF Record 4-37
  - NDT Record 4-37
  - RDT Record 4-37
  - resource access authorizations 4-39
  - resource ownership 4-39, 4-40
  - security environment status 4-41
  - security information 4-36
  - submit ACID information 4-39
- Distributed security 1-3
- Division
  - definition of 1-7
- Divisional Control ACID (VCA) 4-5
- DLFCLASS resource 6-23
- DORMANT Mode 2-3, A-14
- DORMPW suboption 5-14
- DOWN options A-7
- DRC control option 2-5, 3-3
- DRC return codes 11-36
- DSN keyword 6-17, 6-20
- DSNAME keyword
  - PERMIT function 6-10
  - REVOKE function 6-10
- DSNAME resource 6-23
- DSPACE resource 6-23
- DUF
  - See Application interface and/or VSE security interface*
- DUFUPD 11-27
- Dummy facilities 10-4
- Dummy facility
  - modifying 10-5

- DUMP control option A-7

## E

- Emergency ACIDs 2-7
- ENVIRON facility defaults 10-12
- Exclusive LCF 7-40, 7-41, 7-43
- EXIT 8-4
- exit points
  - TERM 11-53
- Expiration interval
  - for existing user 5-6
  - for new user 5-4
  - general explanation 5-2
  - when password does not expire 5-5, 5-7
- EXPIRE operand 4-30, 4-34
- Extending security
  - application interface 11-2
  - site exits 11-47
  - VSE security interface 11-21

## F

- Facilities
  - adding new 10-4
  - associating services with 10-1
  - categories of 10-1
  - default values 10-9
  - definition of 10-1
  - predefined 10-1
  - protecting 10-1
  - tailoring access to 10-1
- Facilities, other 2-20
- FACILITY
  - authorities 4-13
  - control option 3-2, 10-27
  - controlling access 5-17
  - keyword used with LTIME 5-25
  - operands 10-6
  - required administrative authority 5-17
  - resource access restriction 6-29
- Facility definition 10-1
  - changing values
    - general explanation 10-6
    - permanently 10-6, 10-7
    - temporarily 10-7
  - displaying 10-4
- Facility implementation 2-13
- FACILITY keyword 4-13
- Facility Matrix Table 1-20
  - administering 10-6
  - definition of 10-3

Facility Matrix Table (*continued*)  
 displaying contents of 10-4  
 example of 10-4  
 FACILITY control option 10-3, 10-6  
 Facility names, altering 10-8  
 FAIL mode 2-4  
 migration to 2-11  
 FCT keyword 6-17  
 FCT resource 6-23  
 FDT keywords 8-13  
 FDT Record 8-12  
 FDTCODE 8-13, 8-17  
 FDTNAME 8-13, 8-14, 8-16, 8-17  
 FETCH access level 6-31  
 FIELD  
 access levels 7-34  
 Field data  
 extracting 11-15  
 updating 11-15  
 Field Descriptor Table 1-22  
 defining 8-11  
 FIELD keyword 6-17, 7-32  
 FIELD resource 6-23  
 Fixed position masking 6-12  
 Floating pattern masking 6-10  
 FRACHECK 11-21

## G

GAP keyword 6-26  
 GENERIC attribute 6-8, 8-4  
 Generic prefix 6-6  
 Generic prefixing 6-5  
 activating 6-7  
 deactivating 6-7  
 how to specify 6-6  
 in LCF lists 7-43  
 maximum/minimum length 6-6  
 undercutting 6-7  
 using 6-6, 7-5  
 volume 7-5  
 Globally accessible resources 6-25, 6-26  
 Gradual implementation 2-3  
 Group  
 attaching to ACID 4-34  
 expiration 4-34  
 required administrative authority 4-34  
 temporary 4-34  
 GROUP ACIDs 1-7  
 Group, definition of 1-7

## H

Hierarchy, structure of 1-9  
 HSM facility defaults 10-12

## I

IBMFAC keyword 6-18  
 IBMFAC resource 6-23  
 IBMGROUP keyword 6-18  
 IBMGROUP resource 6-23  
 ID  
*See* ACID  
 ID, operator cards 5-26  
 IDMSPROD facility defaults 10-12  
 IDMSTEST facility defaults 10-13  
 IMPLEMENT Mode 2-4  
 Implementation steps 2-1  
 Implementation using modes 2-3  
 Implementing by facility 2-13  
 Implementing CA-Top Secret 2-1  
 Implementing default protection 2-11  
 IMSPROD facility defaults 10-13  
 IMSTEST facility defaults 10-13  
 Inclusive LCF 7-40, 7-41  
 Index substitution 6-11  
 Information repository, user 1-3  
 Installation data, updating 11-14  
 Installation-defined resource 7-32  
 INSTDATA 11-13, 11-26  
 INSTLN operand 11-36  
 INTERACT facility defaults 10-13  
 INTERVAL keyword (MVS)  
 ADD function 8-41  
 REMOVE function 8-41  
 IO exit point 11-54  
 IUCV keyword 6-20  
 IUCV resource 6-23

## J

JCT keyword 6-17  
 JCT resource 6-23  
 JES facility defaults 10-13  
 JESJOBS resource 6-23  
 JESSPOOL resource 6-23  
 Jobs  
 assigning ownership 6-14

## K

Keywords  
 used with the FDT Record 8-13

## L

LCF 7-39  
  bypassing 7-42  
  decision process 7-41  
  exclusive list approach 7-40  
  generic prefixing 7-43  
  implementing 7-39  
  inclusive list approach 7-40  
  LCFCMD 7-39  
  LCFTRAN 7-39  
  mixing LCF types 7-40, 7-41  
  providing default protection 7-42  
  reverification 7-43  
  search levels 7-41  
  setting up 7-40  
  termination of search 7-41  
  uses of 7-39, 7-40  
  using both list types 7-41  
LIB attribute 8-4  
LIB keyword 6-30  
Limited command facility  
  *See* LCF  
Limited Command Facility (LCF) 7-39  
  CMD 7-39  
  TRANS 7-39  
  XCMD 7-39  
  XTRANS 7-39  
Limited Security Control ACID (LSCA) 4-4  
Listing an expiration date for a group 4-34  
Listing an expiration date for a profile 4-30  
Listing, ACID information 4-10  
LOCKTIME keyword 5-25, 5-26  
LOG control option 3-2, 3-3  
Logging  
  security events 3-4  
  using TSSAUDIT 3-4  
  using TSS CPR 3-4  
  using TSSRECVR 3-5  
Logging activity/violations 2-27  
Logon  
  *See* system entry  
LONG attribute 8-4  
LSCA  
  *See* Limited Security Control ACID  
LSCA, determining scope 4-18

## M

Maintaining  
  ACIDs 4-9  
  resources 4-11

Maintenance forms 2-33  
MAP records 8-28  
MASK attribute 6-9, 8-5  
MASK records 8-24  
Masking  
  ACID substitution 6-12  
  combining techniques 6-13  
  fixed position 6-12  
  floating pattern 6-10  
  for data sets 6-9  
  how used 6-9  
  index substitution 6-11  
  options 6-9  
  passwords 5-11  
    character types allowed 5-11  
    defining 5-11  
    explanation of 5-11  
  resources supporting 6-9  
  special considerations 6-13  
  techniques for 6-10  
  to match all data sets 6-11  
  variable character 6-10  
Masking, resources 8-9  
Master ACEE  
  *See* ACEE  
Master Security Control ACID (MSCA) 4-3  
MASTFAC 10-1  
MAXLEN 8-4, 8-13, 8-16  
MERGE 6-38  
MERGE attribute 8-4  
MESSAGE exit point 11-57  
Messages, suppressing 2-28  
MGMTCLAS keyword 6-18  
MGMTCLAS resource 6-23  
MISC1 authority A-10  
MISC1 keyword, authorities 4-14  
MISC2 keyword, authorities 4-15  
MISC3 keyword, authorities 4-16  
MISC8 keyword, authorities 4-16  
MISC9 authority A-8  
MISC9 keyword, authorities 4-17  
Mode  
  password validation, forcing by mode 5-14  
MODE control option A-5  
Model ACID, creating 4-23  
Modes 2-3, 2-5  
Modes, implementing 2-3  
MODIFY(STATUS) 4-41  
MOVE function with TYPE keyword 4-24  
Moving ACIDs with TYPE keyword 4-24

## MSCA

*See* Master Security Control ACID

MSG control option 3-3

Multiple access authorization 6-27, 6-36

Multiple user address space

definition of 10-1

explanation of 11-38

MVS facility

program protection 7-25

resource protection 7-29

sharing Security File with VM 10-28

MVS security interface

multiple user address space 11-38

## N

Native security (CICS, DL/1, IDMS) 2-18

NCCF facility defaults 10-14

NDT Record 5-29

NEWPW control option 5-4, 5-5, 5-16

NOALLMERGE attribute 8-4

Node Descriptor Table 1-22

NODEFPROT attribute 8-4

NODES 6-18

NODSNCHK attribute 7-2, 7-17

NOEXIT attribute 8-4

NOLCFCHK 7-42

NOLCFCHK attribute 7-2

NOLIB attribute 8-4

NOMASK attribute 8-5

NOMERGE attribute 8-4

NONGENERIC attribute 6-8, 6-9, 8-4

how to specify 6-8

overriding 6-9

removing 6-9

NONMASK attribute 6-9

NOPERMIT keyword 6-16

NOPRIVPGM attribute 8-4

NOPW suboption 5-7

NORESCHK attribute 7-2

how to use 7-3

NOSUBCHK attribute 7-2

Notation conventions xx

NOVMDCHK attribute 7-3

NOVMUSER attribute 8-4

NOVOLCHK attribute 7-2

using 7-8

NPWR suboption 5-7

NPWRTHRESH control option 5-7

## O

Odering of profiles 4-31

Online job submission validation 5-27

OPCMD keyword 6-18

OPCMD resource 6-23

OpenMVS Group ACID 1-7

Operator ID cards 5-26

OPERCMD resource 6-23

OTRAN keyword 7-36

OTRAN resource 7-36

OTRAN security

implementing 7-36

setting up 7-37

OVERRIDE 6-37

Ownership 6-15

access level implied 6-13

assigning CPU 6-13

assigning data set 6-14

assigning job 6-14

assigning terminal 6-14

assigning volume 6-14

removing 6-16

## P

PANEL resource 6-23

Parameter file 1-20

PassTickets 5-28

PassTickets, in the NDT Record 5-29

Password

changing 1-12, 5-5, 5-9, 5-12

controlling history of 5-8

defaults 5-3, 5-5

defining 5-4

DORMPW 5-14

encryption 11-35

expiration interval 5-2

expiration intervals 1-12

expiring 5-8

forcing validation by mode 5-14

generating 1-13

masking 5-11

non-expiring 5-5

NOPW suboption 5-7

PASSWORD parameter 5-3

prohibited 5-9

propagation 9-14

PTHRESH control option 5-15

PWHIST control option 5-8

PWVIEW control option 5-8

random generation 5-12

automatic 5-14

Password (*continued*)  
     random generation (*continued*)  
         preparing for 5-13  
         user-requested 5-13  
     restricted list 5-5, 5-9  
     restricting 5-16  
     reverifying changed 5-7  
     RNDPW suboption 5-13  
     rules for creating 5-3  
     setting 5-16  
     validating 1-12  
     viewing 5-8  
     violation threshold 5-15  
         explanation of 5-15  
         setting 5-15  
     WARNPW 5-14  
 PASSWORD exit point 11-54  
 PASSWORD parameter 5-3  
 Password reverification  
     activating 5-7  
     explanation of 5-7  
 PERMIT  
     how to use 6-21  
     with generic prefix 6-21  
 PERSISTV 8-41  
 PGM keyword 6-17, 6-18  
 Phased implementation 2-3  
 POSTINIT exit point 11-53  
 PPT keyword 6-17  
 PPT resource 6-23  
 Prefixed resources B-1  
 Prefixing 6-6  
 Prefixing, generic 6-6  
 PREINIT exit point 11-49  
 Privileged program 5-27, 6-29  
 PRIVPGM attribute 8-4  
 PRIVPGM keyword 6-30  
 Profile  
     attaching to ACID 4-30  
     expiration 4-30  
     Global Profile 4-31  
     ordering of 4-31  
     required administrative authority 4-30  
     temporary 4-30  
 Profile, definition of 1-7  
 PROG keyword 7-25  
 Program  
     accessing all 7-28  
     authorizing access 7-27  
     default protection 7-26  
     defining 7-25  
     generic prefixing 7-26

Program (*continued*)  
     pathing 5-27, 7-7, 7-15, 7-27, 7-35, 7-39  
     privileged 5-27  
     protecting 7-25  
     protection 7-25  
         removing ownership 7-26  
 Program pathing 6-29  
 Program protection  
     assigning default 7-26  
 PROGRAM resource 6-23  
 Program, privileged 6-29  
 Promoting an ACID 4-24  
 PROPCNTL keyword 6-18  
 PROPCNTL resource 6-23  
 Protecting  
     programs 7-25  
 Protecting special resources 2-23  
 PSB keyword 6-17  
 PSB resource 6-23  
 PSFMPL resource 6-23  
 PTHRESH control option 5-15  
 PWHIST control option 5-8  
 PWVIEW control option 5-8

## R

RACDEF 11-21  
 RACF 1-3  
 RACHECK 11-21, 11-23  
     access levels compared 6-32  
 RACINIT 11-21  
 RACROUTE 11-21, 11-22  
 RACROUTE REQUEST=FASTAUTH 11-29  
 RACROUTE REQUEST=VERIFY 11-22  
 RACXTRT 11-21  
 Random password generation 5-12  
 RDT  
     adding resources to 6-2  
     modifying 6-2  
 RDT Record 6-2, 6-4, 6-32, 8-2, 8-3, 8-4, 8-5, 8-6  
 Reader access 5-27  
 Readers, defining 7-23  
 RECID resource 6-23  
 Record Level Protection  
     description of  
 Records  
     Resource Descriptor Table (RDT) 6-2  
 RECOVER control option A-8  
 Recovery  
     using TSSRECVR 3-5  
 Recovery file 1-21, 3-5, A-20

- Recovery procedures
  - Recovery File A-8
- Reporting activity/violations 2-27
- Reports
  - running 3-6
  - TSSCFILE option 3-7
  - TSSCHART option 3-6
  - TSSREPORT option 3-7
  - TSSREPORT2 option 3-7
  - TSSREPT option 1 3-7
  - TSSREPT option 2 3-7
  - TSSUTIL option 3-6
- Request fields 11-4
- Request record 11-3
- Required administrative authority 4-19
- RESCLASS 8-3, 8-7, 8-8, 8-9
- RESCODE 8-3
- RESETEOD control option A-15
- RESOURCE
  - authorities 4-11
  - authorizing 6-1
  - authorizing access 6-24
  - bypassing 7-2
  - bypassing security checking 7-3
  - checking 7-2
  - controlling access to 6-21
  - default protection 6-3, 6-22
  - defining 6-1, 6-3
    - by full name 6-5
    - by generic prefix 6-5
  - denying access 6-34
  - determining ownership 6-15
  - facility restriction 6-29
  - generic prefix support 6-7
  - globally accessible 6-25, 6-26
  - how to protect 7-2
  - installation defined 7-32
    - assigning access levels 7-34
    - authorizing access 7-34
    - generic prefixing 7-33
    - pathing 7-35
    - removing ownership 7-33
  - keyword 6-3
  - owner, locating 6-15
  - ownership
    - assigning 6-4
    - by Department, Division or Zone 6-4
    - by Users 6-4
    - general description 6-3
    - reducing data entry 6-5
    - splitting between Departments 6-5
    - transferring 6-15

- RESOURCE (*continued*)
  - predefined classes
    - MVS and/or VSE 6-17
    - VM 6-20
    - VSE 6-19
  - process for securing 6-1
  - requiring administrative authority 6-3
  - securing 6-1
  - selecting owners 6-4
  - time and duration restrictions 5-18, 6-24
  - transferring ownership 6-15
  - VM protection 6-1
  - VSE protection 6-1
- Resource checking
  - bypassing 7-2
- Resource class 6-3
- Resource Descriptor Table 1-22
- Resource Descriptor Table (RDT) 6-2
  - customizing 6-2
  - viewing contents 6-2
- RESOURCE exit point 11-51
- RESOURCE keyword, operands 4-11
- Resource masking 8-9
- Resource name list service support 11-45
  - invoking support 11-45
  - return codes 11-46
  - returned format 11-46
- Resource ownership 6-15
- Resource processing 6-40
- Resource protection 1-3
  - bypassing 7-2
  - commands 7-1
  - facilities 7-1
  - for MVS 7-29
  - procedure for 7-2
  - resources 7-1
  - types protected 7-1
- Resources
  - prefixed B-1
- Restricted password list 5-3
  - activating 5-10
  - adding entry to 5-10
  - changing from operator console 5-10
  - default entries 5-10
  - displaying contents of 5-11
  - general explanation 5-9
  - maximum entry reached 5-11
  - removing entry from 5-11
- Restricting a facility 6-29
- Restricting signons 5-21

- Retrieving an ACID 11-16
- Return codes 11-7
- Reverification 7-43
- RLP
  - See* Record Level Protection
- RLP records 8-25
- RNDPW suboption 5-13
- ROSCOE facility defaults 10-14

**S**

- Sample code 11-17, 11-19
- SCA
  - See* Security Control ACID
- SCOPE keyword, authorities 4-18
- Screen Level Protection
  - description of
- SDSF resource 6-23
- SDT Record 8-19
- SECTRACE control option A-8
- Secured Signon support 5-28
- Securing
  - individual resources 6-4
  - online applications 7-29
  - owned transactions 7-36
  - programs in MVS 7-25
  - resources 6-1
    - by default 6-22
  - unownable resources 7-39
- Security
  - creating your hierarchy 1-10
  - distributing 1-4
  - for system entry 5-17
  - hierarchy 1-9
  - information, displaying 4-36
  - logging events 3-4
  - modes
    - concurrent 2-5
  - reports, batch utility option 3-6
- security administrator 1-11
  - role of 1-10
- Security Control ACID (SCA) 4-4
- Security Features
  - BATCH 10-23
  - CA-IDMS 10-20, 10-22
  - CICS 10-17, 10-19
  - VM 10-27
- Security file 1-20
  - monitoring changes 3-4
  - sharing between VSE, VM and MVS 10-28
- Security File backup
  - automatic backup feature A-18

- Security File recovery
  - manual backup A-19
- Security File serialization A-21
- Security file, maintaining 2-33
- Security interfaces 1-4
- Security key, CICS 10-19
- Security Record 4-19
  - displaying ACID information 4-10
- Security validation algorithm 6-36, 6-37, 6-38
  - AUTH control option 6-37
  - best fit, determining 6-46
  - concept of 6-38, 6-44
  - controlling search 6-40
  - data set call 6-43
  - data set checking 6-42
  - general resource checking 6-45
  - how it works 6-36
  - setting up 6-37
  - volume checking 6-40
- SEGMENT 8-13, 8-15, 8-18
- SELECT records 8-26, 8-29
- SESEND exit point 11-55
- SESSKEY keyword
  - used with LINKID 8-41
- SESSKEY keyword (MVS)
  - ADD function 8-41
  - REMOVE function 8-41
  - REPLACE function 8-41
- SFSCMD resource 6-23
- SHORT attribute 8-4
- Shutdown procedures
  - end-of-day A-14
  - temporary shutdown A-14
- Signon
  - See* system entry
- Signons, restricting 5-21
- Single user address space
  - definition of 10-1
- Single vs. multiple user address space 10-1
- SITE exit point 11-59
- Site exits
  - activating the exit 11-61
  - activation matrix 11-48
  - application interface 11-2
  - entry points
    - RESOURCE 11-51
  - exit points 11-62
    - ACTION 11-57
    - CHANGE 11-56
    - COMMAND 11-51
    - CPF 11-59
    - DATASET 11-50
    - IO 11-54

Site exits (*continued*)

exit points (*continued*)

MESSAGE 11-57  
PASSWORD 11-54  
POSTINIT 11-53  
PREINIT 11-49  
SESSEND 11-55  
SITE 11-59  
SUBMIT 11-55  
UNDEFIND 11-54  
VIOLATN 11-58  
VOLUME 11-50

EXIT(OFF) control option 11-61

EXIT(ON) control option 11-61

extending security 11-2

ideas for customizing 11-48

sample code 11-48

TSSINSTX 11-48, 11-60

user entry points 11-62

ACTION 11-57

CHANGE 11-56

COMMAND 11-51

CPF 11-59

DATASET 11-50

IO 11-54

MESSAGE 11-57

PASSWORD 11-54

POSTINIT 11-53

PREINIT 11-49

RESOURCE 11-51

SESSEND 11-55

SITE 11-59

SUBMIT 11-55

UNDEFIND 11-54

VIOLATN 11-58

VOLUME 11-50

uses of 11-48

SLP

*See* Screen Level Protection

SMESSAGE resource 6-23

Software maintenance, system 2-35

Software maintenance, TSS 2-34

source of origin 5-22

SOURCE parameter 5-22

Special authentication devices

image verification 5-26

voice verification 5-26

SPI keyword 6-18

SPI resource 6-23

START TSS A-13

Startup procedures

activating TSS A-13

STATUS control option A-10

STC facility defaults 10-14

STORCLAS resource 6-23

STORCLASS keyword 6-18

SUBMIT exit point 11-55

SUBSCHEM keyword 6-18

SUBSCHEM resource 6-23

SUSPEND control option A-10

SYNCH control option A-10

Syntax

description of components 1-17

SYSCONS resource 6-23

SYSOUT control option A-11

System Authorization Facility 7-29, 11-21

System entry

batch job 5-27

CPU 5-27

facility 5-17

image verification 5-26

online job submission 5-27

protection 5-17

reader 5-27

securing 1-6

terminal 5-24, 5-27

voice verification 5-26

System entry validation 1-3

with CPF 9-14

System entry, automatic 5-23

System entry, default ACID 5-23

## T

Tape

external management 2-21

management packages 7-18

protection 2-21

Tape bypass label processing 7-19

TAPE control option 7-18

Tape volume

protecting 7-18

TARGET keyword 9-7

Temporary groups 4-34

Temporary profiles 4-30

TERM exit point 11-53

TERM keyword 6-17, 6-20

Terminal

access 5-27

accessing all 7-24

assigning ownership 6-14

automatic locking 5-24



Terminal (*continued*)

- controlling access 5-21
- defining 7-21
- generic prefixing 7-22
- locking 5-24
- LTIME 5-25
- manual terminal locking 5-26
- PC definitions 7-22
- protecting 7-21
- removing ownership 7-21
- restricting access 7-23
- source of origin security 5-21
- unlocking 5-26
- VM definitions 7-22

Terminal protection

- types of 7-21

TERMINAL resource 6-23

Terminals, security for 2-23

TIME records 8-27

TONE facility defaults 10-14

TRACE attribute A-8

TRAN keyword 6-18

TRANS keyword 7-39

Transaction security 7-36

- See also* LCF

Transactions

- accessing all 7-43
- authorizing access 7-38
- generic prefixing 7-38
- how secured 7-36
- LCF security 7-39
- OTRAN security 7-36
- program pathing 7-39
- removing ownership 7-38

TSO facility defaults 10-14

TSOACCT resource 6-23

TSOACT keyword 6-18

TSOAUTH keyword 6-18

TSOAUTH resource 6-23

TSOPRFG keyword 6-18

TSOPRFG resource 6-23

TSOPROC keyword 6-18

TSOPROC resource 6-23

TSS automatic backup feature A-6

TSS Command Functions

- components 1-17

TSS commands

- LIST 4-36, 4-37, 4-38
- MODIFY 10-3, 10-6, 10-28
- MODIFY(STATUS) 4-41
- PERMIT 6-21
- WHOAMI 4-41

TSS commands (*continued*)

- WHOHAS 4-39
- WHOOWNS 4-40

TSS interfaces 2-21

TSS MODIFY command A-11

TSS MOVE command function with TYPE keyword 4-24

TSS restart A-16

TSSAI

- See* application interface

TSSAUDIT 3-4, 3-8

TSSBACKUP A-19

TSSBACKUP procedure A-19

TSSCFILE 3-7

TSSCHART 3-6

TSSCPR 3-4

TSSCPR utility 9-14

TSSMAINT utility A-17, A-20

TSSRECVR 3-5

TSSRECVR utility A-20

TSSREPORT 3-7

TSSREPORT2 3-7

TSSREPT

- option 1 3-7
- option 2 3-7

TSSTRACK utility 2-28

TSSUTIL 3-6

TST keyword 6-17

TST resource 6-23

TYPE keyword used with TSS MOVE 4-24

TZONE 5-20

## U

UNDEFIND exit point 11-54

UNDERCUT keyword 6-7, 6-15

Undercutting 6-7, 6-15

- rules for 6-7
- to transfer ownership 6-15

Unlocking a terminal 5-26

Unownable resources

- securing through LCF 7-39
- securing through OTRAN 7-36

UR1 keyword 6-17, 7-32

UR1 resource 6-23

UR1/UR2

- access levels 7-34

UR2 keyword 6-17, 7-32

UR2 resource 6-24

User 6-18

- definition of 1-7

User authentication 5-26

USER C/B  
     *See* ACEE  
 user entry points  
     TERM 11-53  
 User information repository 1-3  
 User-oriented architecture 1-3  
 USER0 facility defaults 10-15  
 USER221 facility defaults 10-15  
 USERnnn facility 10-5  
 USERx 7-32  
 USING keyword 4-23  
 USRCLASS resource 6-24

## V

VAMSPF facility defaults 10-15  
 Variable character substitution 6-10  
 VCA  
     *See* Divisional Control ACID  
 VERSION control option A-11  
 Violation threshold, password 5-15  
 Violations 3-2  
 VIOLATN exit point 11-58  
 VM facility  
     activating 10-28  
     authorizing 10-27  
     deactivating 10-28  
     protected resources 6-1  
     sharing Security File with MVS 10-28  
 VM facility defaults 10-15  
 VMCF keyword 6-20  
 VMCF resource 6-24  
 VMDIAL keyword 6-20  
 VMDIAL resource 6-24  
 VMFAC control option 10-27  
 VMMACH keyword 6-20  
 VMMACH resource 6-24  
 VMMDISK keyword 6-20  
 VMMDISK resource 6-24  
 VMNODE keyword 6-20  
 VMNODE resource 6-24  
 VMRDR keyword 6-20  
 VMRDR resource 6-24  
 VMUSER attribute 8-4  
 VOL keyword 6-17  
 Volume  
     access levels 7-6  
     assigning ownership 6-14  
     authorizing access 7-5  
     backing up 7-6, 7-9  
     bypassing data set security 7-8  
     bypassing security checking 7-8

Volume (*continued*)  
     bypassing security for 7-8  
     compacting 7-9  
     data set, versus 7-20  
     default protection 7-5  
     generic prefixing 7-5  
     how to access all 7-7  
     management packages 7-9, 7-18  
     NOCREATE 7-6  
     program pathing 7-7  
     required administrative authority 7-8  
     restoring 7-6, 7-9  
     securing relations to data set security 6-42  
     tape 7-18  
 VOLUME exit point 11-50  
 VOLUME keyword 6-20  
 Volume protection  
     access request 7-4  
     bypassing 7-8  
     establishing ownership 7-4  
     extending default 7-5  
     removing ownership 7-4  
 VOLUME resource 6-24  
 Volume/Data set access authorization relationships 6-42  
 VSE facility  
     protected resources 6-1  
     sharing Security File with MVS 10-28  
 VSE security interface  
     DUFXTTR 11-26  
     Dynamic Update Facility (DUF) 11-26  
         error return codes 11-29  
         tips for using 11-29  
     FRACHECK 11-21  
     information feedback 11-36  
     INSTDATA 11-26  
     RACDEF 11-21  
     RACHECK 11-21, 11-23  
     RACINIT 11-21  
     RACROUTE 11-21, 11-22  
     RACROUTE REQUEST=AUTH 11-23  
     RACROUTE REQUEST=AUTH class names 11-24  
     RACROUTE REQUEST=EXTRACT 11-34  
         encryption 11-35  
         obtaining feedback 11-35  
     RACROUTE REQUEST=FASTAUTH 11-29  
         character string class name, specifying 11-30  
         class name formats 11-30  
         class name list 11-31  
         performance shortcuts 11-33  
         return codes 11-33  
         specification 11-34  
     RACROUTE REQUEST=VERIFY 11-22

VSE security interface (*continued*)  
RACXTRT 11-21  
resource name list service support 11-45  
SAF, use of 11-21  
VSELIB keyword 6-19  
VSEMEMBR keyword 6-20  
VSEPART keyword 6-20  
VSESLIB keyword 6-20  
VTAMAPPL keyword 6-18  
VTAMAPPL resource 6-24  
VTHRESH control option 3-2, 3-3  
VXDEVICE resource 6-24  
VXFILE resource 6-24

## W

WAIT keyword 9-7  
WARN Mode 2-3  
WARNPW suboption 5-14  
WHOOWNS 6-15  
WRITER resource 6-24  
WYLBUR facility defaults 10-16

## X

XCMD keyword 6-18, 7-39  
XTRAN keyword 6-18  
XTRANS keyword 7-39

## Z

ZCA  
*See* Zonal Security Control ACID  
Zonal Security Control ACID (ZCA) 4-5  
Zone  
definition of 1-8



# User Registration Form

---

If you are interested in registering as a user of Computer Associates software, please complete the information below. Send it to the following address:

Computer Associates International, Inc.  
ATTN: User Registration  
One Computer Associates Plaza  
Islandia, NY 11749

Registration entitles you to receive such items as:

- newsletters
- product release announcements
- information about User Groups
- information about CA conferences
- client questionnaires

You *do not* have to be the primary contact for CA software within your company or organization. All you have to be is interested in CA software, how it is used, and where it is headed.

Product(s): \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

Site ID: \_\_\_\_\_  
(Enter UNKNOWN if you do not know your Site ID.)

Name: \_\_\_\_\_

Position: \_\_\_\_\_

Company or organization: \_\_\_\_\_

Address: \_\_\_\_\_

Address: \_\_\_\_\_

Phone No.: \_\_\_\_\_

Date: \_\_\_\_\_

I would like additional information on: \_\_\_\_\_









# Reader Comment Form

---

CA-Top Secret User Guide

Release 3.0 VSE

Document Number: R101TS30UGE

Please use this form to tell us how you use this manual and to make suggestions for improvement. Send it to the following address. (To request additional publications, call 1-800-841-8743 or check the CA home page (<http://www.cai.com>) on the Internet. To ask questions about the functionality of CA products, contact your CA representative.)

Computer Associates International, Inc.  
ATTN: Reader Comment Form  
One Computer Associates Plaza  
Islandia, NY 11749

Be sure to print your name and address below if you would like a reply.

Name: \_\_\_\_\_

Position: \_\_\_\_\_

Company or organization: \_\_\_\_\_

Address: \_\_\_\_\_

Address: \_\_\_\_\_

Phone No.: \_\_\_\_\_

Date: \_\_\_\_\_

Years of experience with this CA product: \_\_\_\_\_

## Overall Rating

|              | Good | Fair | Poor | Why? |
|--------------|------|------|------|------|
| Accuracy     |      |      |      |      |
| Organization |      |      |      |      |
| Clarity      |      |      |      |      |

## Reference Aids

|                            | Good | Fair | Poor | Why? |
|----------------------------|------|------|------|------|
| Contents                   |      |      |      |      |
| Index                      |      |      |      |      |
| Glossary                   |      |      |      |      |
| Reference to Other Manuals |      |      |      |      |

## Clarity

|                             | Good | Fair | Poor | Why? |
|-----------------------------|------|------|------|------|
| Instructions and Procedures |      |      |      |      |
| Examples                    |      |      |      |      |
| Graphics                    |      |      |      |      |

**How Manual Is Used:**

How do you use this manual in your job?

How often do you use this manual in a week?

**Suggestions:**

What do you like about this manual?

What do you dislike about this manual?

What would you like us to change?

**Additional Comments:**

---

---

---

---

---

---

---

---

---

---

