CA Top Secret® for z/VM

Customization Guide

r12

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: Introduction

This section contains the following topics:

Customization In General

This guide, intended primarily for the systems programmer, describes the features within CA Top Secret that allow for the tailoring of security and special requirements at your installation.

You can extend CA Top Secret through user exits and site-written code to enforce security restrictions unique to your environment.

Although CA Top Secret has tremendous flexibility in meeting the security requirements of most installations, your installation may face a requirement that cannot be accommodated by CA Top Secret as delivered on the installation tape. In this situation, customization may be appropriate.

Common Reasons For Customization

Customization is most often used in the following situations

Virtual Machine Interfaces

CA Top Secret provides an Application Program Interface containing installation-written code with the ability to issue security calls.

CA Top Secret Installation Exit

Customization can be used to modify CA Top Secret behavior to meet special customer requirements through the CA Top Secret Installation Exit.

The Installation Exit provides initiation, validation, logging, message,
CA Top Secret Security File change, and other exit points for user routines.
Customization at this level has been successfully done to add features to CA Top Secret.
The exit has been used to:

- Translate messages into different languages
- Keep multiple CA Top Secret Security Files in sync across CPUs

  Note: This function can now be done using the Command Propagation Facility.

- Provide interfaces for second level authentication
- Establish the ACID that is used at logon.

Considerations Before Customization

- Customization using the Installation Exit is not limited to the examples detailed. However, it is strongly recommended that the Installation Exit not be used to bypass security, or to change CA Top Secret behavior to something that does not complement documented CA Top Secret features.
- Avoid customization through any other means than those detailed above. CA is committed to support only the CA Top Secret capabilities.
- Many CA Top Secret customers have successfully customized their use of CA Top Secret through the available features. However, you should exercise careful analysis and planning, including research of existing CA Top Secret features, before deciding to customize. Often a reevaluation of the desired approach eliminates a customization requirement.

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, contained in module TSSINS.

CA Top Secret offers alternate methods of customization, such as the Application Interface. Before committing your installation to a particular type of customization, refer to the *Planning Guide*.

Customization Ideas

A Sample Installation Exit is provided on the distribution tape as file TSSINSV1 ASSEMBLE. It includes sample code for a variety of uses of the Installation Exit including:

- Set up standard ACID ID name
- Set up standard undefined ACID ID in warn or dorm mode
- Prevent the use of any three consecutive characters in a new password
- Restrict access of write or update to certain restricted dataset if on the proper volumes
- Restrict use of permit mode

Mechanics

Entry point TSSINSTX is called by CA Top Secret at various points during security processing. A function code indicates the type of event being processed and the purpose of the exit call. This code is contained in the passed parameter list called the TXIPL (mapped by the #INSTXPL macro). The TXIPL also provides additional fields to describe the event, many of which can be modified or overridden by the exit if desired.

Prior to CA Top Secret's request processing, the data is examined by the Installation Exit. Upon completion, the Installation exit supplies a return code to CA Top Secret. The return code determines the next course of action.

All of the return codes are documented in TSSINSV1. Those common to all exit functions are:

Return Code	Action
0	Allows CA Top Secret to perform the normal security request authorization.
4	The request is failed immediately by the Installation Exit code. CA Top Secret does NOT process the request.
8	The Installation Exit authorizes the request. CA Top Secret's equest validation will be superceded (ignored).

TSSINSTX Characteristics

TSSINS is a single object module with one entry point, TSSINSTX. It resides in the CA Top Secret server nucleus and must be re-entrant.

The Installation Exit is entered in virtual supervisor state in the storage key of the security kernel. The exit must not perform any I/O or invoke security checks.

The Installation Exit is protected by an error recovery routine. In the event of an abend, a SVC dump is taken and the exit is disabled with a message issued to the system operator.

Activating The Installation Exit

The CA Top Secret EXIT control option controls the activation or deactivation of the Installation Exit.

EXIT(ON)

This setting causes CA Top Secret to interact with the installation exit module, TSSINSTX.

EXIT(OFF)

This setting deactivates Installation Exit calls.

In order to activate a new exit, the server nucleus must be rebuilt and the CA Top Secret server must be restarted.

Chapter 2: Using the Application Interface

This section contains the following topics:

About the Interface

The CA Top Secret Application Interface provides a mechanism by which application programs may invoke the services of the CA Top Secret VM server machine. These services include resource validation, dynamic extract and update of Security Record installation data, password extraction, logging of user data, and establishing a security environment.

The Application Interface is invoked using the security diagnose (X'A0') with the Rx register containing the address of the 572-byte application interface parameter list (AIPL) and the Ry register containing a subcode of 3. The AIPL must be aligned on a doubleword boundary, and the first 16 bytes of the AIPL must be contained within the same 4K page of storage or a specification exception results. A condition code of 0 from the diagnose indicates that the request was processed by the server machine. The function code (AIFUNC), return code (AIRC), detail reason code (AIDRC), and messages (AI#MSG for message count, AIMSG for message text) are returned in the AIPL. A condition code of 1 indicates that the server machine was unavailable for the request. In some cases, a valid return code is still provided. (See the individual function descriptions below.) The AIPL field requirements and restrictions are as follows:

Resource Validation

In order to validate a resource, the following fields in the AIPL must be supplied:

AIFUNC

Contains AI$RES

AIREST

Contains the hex resource code as defined in the RDT for the resource being checked. (CA-defined resources are found in the #AFLAGS macro in TSSVM MACLIB.)

AIREST2

Contains the two-byte resource code as defined in the RDT for he resource being checked. If a two-byte resource code is being used, AIREST must be set to x'FF' to signify the fact.

AIRESN

Contains the name of the resource to be validated.

Note: If you are indicating the name of an SFS resource (such as SFSCMD or DIRECTRY), the resource name cannot exceed 44 characters in length.

AIRESL

Contains the MACHINE LENGTH (actual length - 1) of the resource being validated

AIRRACC

Contains composite one byte result of all access levels required to satisfy the request

AIRRACCE

Contains composite two byte result of all extended access levels required to satisfy the request (see note below).

AIVOL

Optionally contains the VOLSER if the resource being validated is an O/S data set

AIUCLASS

Contains the class for USERx validation

Note: Your application may use either AIRRACC or AIRRACCE, but not both. AIRRACC is maintained for existing applications that were written with support for one byte access levels. All new applications, or existing applications that require the two byte expanded access levels, should set AI$EXT in AIFLAG1 and use AIRRACCE instead.

Additionally, the following bit settings for AIFLAG1 are honored:

AIF$NLOG

Do not log this resource validation request

AIF$PRIV

The validation is being done for the privileged form of a CP command or diagnose.

AIF$VMID

AIACID contains a VM userid

AIF$MUM

An AI$INIT function was previously done for this ACID.

AIF$EXT

AIPL contains "extended access levels" in AIRRACCE.

If the resource is being validated on behalf of a logged on virtual machine, the AIACID field should contain the name of that virtual machine and AIF$VMID should be set in AIFLAG1. If the AIACID field is not filled in, the validation is processed against the issuer of the diagnose.

If the request is being done on behalf of another virtual machine, and the issuer has the MASTFAC attribute, the check is made under the restrictions (mode, facility, etc.) of the MASTFAC facility.

For more details on the use of MASTFAC, refer to the *Command Functions Guide*.

Other than access to the security diagnose (X'A0'), issuer needs no special authority from CA Top Secret in order to validate the resource access.

Return Codes

The return codes contained in AIRC are as follows:

AIRC = 0

Resource access allowed.

AIRC = 4

Resource access deferred - CA Top Secret has neither permitted nor denied access to the resource.

AIRC = 8

Resource access not allowed.

AIRC = 12

Resource access not allowed - Violation threshold exceeded.

AIRC = 16

Syntax error in AIPL.

If the AIRC is not zero, the AIDRC field contains the detail error reason code indicating why the resource access is not allowed. The detail error reason codes are documented in the #DFLAGS macro. If there are messages to be issued, AI#MSG contains the message count and AIMSG contains the message(s).

An Application Interface call of this type always returns a valid AIRC. If the server is unavailable, condition code 1 is set as a result of the diagnose, and the AIRC field contains 0, 4, or 8 depending on the DOWN option. If CA Top Secret has not been initialized since the most recent VM IPL, this diagnose function results in a specification exception.

Dynamic Extract

In order to extract an ACIDs installation data, the issuer of the diagnose must have the DUFXTR attribute. The following fields in the AIPL must be supplied:

AIFUNC

Contains AI$DUFX

AIACID

Contains the name of the ACID from which to extract installation data.

The return codes contained in AIRC are as follows:

AIRC = 0

Extract successful - AIINST contains a one byte length followed by the installation data

AIRC = 4

The ACID specified in AIACID has no INSTDATA

AIRC = 8

Error extracting installation data - See AIDRC

AIRC = 12

Caller does not have DUFXTR authority

AIRC = 16

Syntax error in AIPL

The installation data is returned in the AIINSTD field of AIPL, and the length of data is returned in the AIINSTL field.

Dynamic Update

In order to update an ACIDs installation data, the issuer of the diagnose must have the DUFUPD attribute. The following fields in the AIPL must be supplied:

AIFUNC

Contains AI$DUFU.

AIACID

Contains the name of the ACID whose installation data is to be updated.

AIINSTL

Contains the MACHINE LENGTH (actual length - 1) of the installation data.

AIINSTD

Contains the installation data itself.

The return codes contained in AIRC are as follows:

AIRC = 0

Update successful.

AIRC = 4

The ACID specified in AIACID has no INSTDATA.

AIRC = 8

Error updating installation data - See AIDRC.

AIRC = 12

Caller does not have DUFUPD authority.

AIRC = 16

Syntax error in AIPL.

Password Extract

This function is useful for a virtual machine through which batch jobs can be routed to a guest or external operating system for execution on behalf of other users. Care should be taken to adequately protect this machine and to destroy any record of these passwords once used.

In order to extract a password for a given ACID, the issuer of the diagnose must have been permitted the use of diagnose X'00A0' with ACTION(VMPRIV), and either the use of the other ACID or have the NOSUBCHK attribute. Under no circumstances is the ACID allowed to extract its own password. The following fields in the AIPL must be supplied:

AIFUNC

Contains AI$PWX.

AIACID

Contains the name of the ACID from which to extract the password.

The return codes contained in AIRC are as follows:

AIRC = 0

Extract successful.

AIRC = 4

The ACID specified in AIACID has no password.

AIRC = 8

Error extracting password - See AIDRC.

AIRC = 12

Caller does not have authority to extract the ACID's password.

AIRC = 16

Syntax error in AIPL..

If successful, the plaintext password is returned in the AIPW field (padded to the right with blanks).

Logging of User Data

CA Top Secret provides a mechanism by which user data can be logged to the Audit File. To accomplish this, the following fields in the AIPL must be supplied:

AIFUNC

Contains AI$LOG.

AILOGL

Contains the MACHINE LENGTH (actual length - 1) of the data to be logged.

AILOGD

Contains the data to be logged

Additionally, the following setting of AIFLAG1 is honored:

AIF$SUSP

Suspend the current ACID of the issuer.

The log entry is recorded using the ACID and VM userid of the issuer of the diagnose.

ACID Extract

This function is used to obtain the ACID of a logged on user. The following fields in the AIPL must be supplied:

AIFUNC

Contains AI$AIDX

AIACID

Contains the userid from which to obtain the ACID.

The return codes contained in AIRC are as follows:

AIRC = 0

Extract successful, AIACID contains the requested ACID.

AIRC = 8

The userid specified in AIACID was not logged on.

AIRC = 16

Syntax error in AIPL.

ACID Verification

This function is used to verify that a specified ACID has been defined to the security file. In order to use this function you must specify OPTIONS(18) in the start up parms. The following fields in the AIPL must be supplied:

AIFUNC

Contains AI$RES.

AIREST

Contains $ARACID.

AIRESN

Contains the ACID name to verify, padded with blanks to 8 characters.

AIRESL

Must contain the value “7”.

The return codes contained in AIRC are as follows:

AIRC = 0

The specified ACID does exist in the database.

AIRC = 8

The specified ACID does not exist in the database.

AIRC = 16

Parameter list error or OPTIONS(18) is not set.

Field Information Extract

This function is useful for extracting information in user-defined fields on an ACID. In order to extract the information for a given ACID, the one issuing the diagnose must have been permitted use of the other ACID or have the NOSUBCHK attribute.

AIFUNC

Contains AI$FDTUX

AIFDTNME

Contains the field name to be extracted

AIACID

Name of ACID containing the field (leave blank for self)

AIFDTOUT

Address of user supplied buffer to contain extracted information

AIFDTLN1

Length of buffer in AIFDTOUT

AIFDTCODE

Returned FDTCODE used by system

AIFDTLN2

Size of actual data in user ACID

AIFDTLN3

Maximum field allowed as defined in FDT

The return codes contained in AIRC are as follows:

AIRC = 0

Extract successful

AIRC = 4

ACID does not have any FDT fields attached

AIRC = 8

ACID has FDT fields attached, but not the requested field

AIRC = 12

Bad parameters passed

AIRC = 16

Caller doesn't have necessary authority

Chapter 3: Establishing a Multiple ACID Environment

This section contains the following topics:

Multiple ACID Environments

For the server machine which may be processing a variety of security requests for multiple virtual machines, it may be advantageous to establish a separate security environment for each ACID. The multiple ACID INIT function creates a logged-on Security Record for the target ACID and associates that record with the issuer of the function. By maintaining logged-on ACID records for each user actively being serviced, overhead can be dramatically reduced for frequent AI$RES calls. Also, if this server machine provides for some sort of logon validation mechanism, the validation may be processed by CA Top Secret through the application Interface diagnose.

Logon Validation

In order to validate a logon or establish a security environment for an ACID, the following AIPL fields are required:

AIFUNC

Contains AI$INIT

A**IACID**

Contains the name of the ACID to be logged on

AINPW

Contains the new password to be used for this user

AIPW

Contains the ACID's CA Top Secret password

AITRMID

Contains terminal address to be used as part of INIT verification

The following flag setting of AIFLAG1 can also be used:

AIF$NOPW

Logon is performed bypassing password validation (that is, the AIPW field is not validated)

AIF$PWRV

The calling program has re-verified the new password

Note: If the control option is set to require a new password be re-verified, the calling program must do that verification and set the AIF$PWRV flag to inform CA Top Secret that the verification has been done.

In order to issue an AI$INIT, the issuer of the diagnose must be authorized to diagnose X'A0' with the VMPRIV attribute.

If the issuer of the diagnose has a MASTFAC (master facility) associated with it, all validation (facility, resource, etc.) is done using that facility restriction. If the user has no associated MASTFAC, the validation requests are done using FAC(VM).

Return Codes

The return codes contained in AIRC are as follows:

AIRC = 0

INIT successful

AIRC = 4

ACID not defined to security

AIRC = 8

INIT failed - See AIDRC

AIRC = 16

Syntax error in AIPL

If a password change was requested, it is possible to receive an AIRC = 0 and have the password change fail. If the new password is invalid, the reason could be that it is too soon to change the password, then AIRC will be set to zero and the INIT will have happened. The caller should verify that AIDRC is also zero. A non-zero value can be found in #DFLAGS and will have messages returned (AI#MSG and AIMSG) describing the returned DRC.

Session Termination

When a user logs off a multiple ACID application or a disconnected server machine has completed its services for a given user, the multiple ACID application uses the session termination function to "clean up" the user environment. The termination function frees the storage allocated in the CA Top Secret server module which contains the user's Security Record.

For removing the security environment established by the AI$INIT function, a converse function, AI$TERM, is provided. The following fields in the AIPL must be supplied:

AIFUNC

Contains AI$TERM

AIACID

Contains the name of the ACID whose session is to be discontinued.

Return Codes

The return codes contained in AIRC are as follows:

AIRC = 0

Session termination successful

Chapter 4: Security Diagnose Subfunctions

This section contains the following topics:

The Surrogate Function

The surrogate function in CA Top Secret offers additional flexibility in extending users' security specifications to surrogate virtual machines.

Surrogacy provides a mechanism by which an authorized user may dynamically assign the authorities of specific ACIDs to virtual machines. In CA Top Secret, a virtual machine can assume the authorities of an alternate ACID -- temporarily overriding the authorities which have normally been established at logon for the virtual machine. For example, a VM batch master control machine may activate a worker machine called WRK7 with USERA's ACID when USERA requires that machine for job execution. Upon job completion, WRK7 is either reset to its original ACID or set to another submitter's ACID by the batch master control machine. At all times USERA, if logged on, always retains access to its resources.

By supporting the use of alternate ACIDs for virtual machines, CA Top Secret keeps your data secure whenever tasks are assigned to worker machines. Security violations can be traced to either the submitting ACID or the worker machine.

See the *Implementation Guide* for information on surrogate control.

Alternate Userid Diagnose

The Alternate Userid Diagnose (X'D4') sets and resets the identity of the ACID on whose behalf the worker machine is executing.

The Alternate Userid Diagnose function is defined as follows:

Rx = Binary zero

Ry = Virtual address of a 16-byte parameter list. It must not cross a page boundary.

Authorization to execute Alternate Userid Diagnose functions requires CA Top Secret permission to DIAG(D4). The Alternate Userid Diagnose requires VMMACH(workerid) ACC(SUROGATE) and ACID(alternate-acid) privileges.

For a complete description of the use of surrogate functions, refer to the discussion of the SUROGATE command in the *Implementation Guide*.

Note: The DIAG(D4) will not give a return code of 12 if the target ACID is not defined, even if the issuer has NOSUBCHK on their ACID.

Security Diagnose - Logon Password Verification

The Security Diagnose (X'A0') provides Logon Password Verification (subcode X'04'). This diagnose is used by application programs or multiuser service machines to verify the identity of the user. (Password verification refers to the user, not the issuer.)

This diagnose verifies the appropriate password required by the user to log on to the system (not specifically CA Top Secret or CP Directory), and is dependent upon:

- User type
- Mode
- Cacility
- Control options (DORMPW, WARNPW)
- Installation Exit

Password violation counts are controlled by PTHRESH if this function is executed with an incorrect password.

The Logon Password Verification function is defined as follows:

CA Top Secret permission to DIAG(A0) ACTION(VMPRIV)

Rx = Address of parameter list. It must be double word aligned and cannot cross a page boundary.

Ry = X'04' subcode

Logon Password Verification Parameter List		
Position	Length	Explanation
0(0)	8	The target userid of the password query
8(8)	8	The password that is to be verified

Logon Password Verification Return Codes	
X'04'	ACID not defined to CA Top Secret (FAIL mode) or user not defined in CP Directory (WARN/DORM mode with NOWARNPW/NODORMPW attribute only)
X'08'	Password invalid
X'0C	Password correct, but expired
X'18	Failed by Installation Exit
X'1C	ACID suspended, expired, or inactive
X'20'	CA Top Secret inactive

Security Diagnose - Return CA Top Secret Server Userid

The Return CA Top Secret Server Userid function may be used to determine whether CA Top Secret is active in the system, and if so, obtain the VM userid of the server.

The Return CA Top Secret Server Userid function is defined as follows:

- Rx = First (even) register of an even/odd register pair.
- Ry = X'07' subcode

To issue the diagnose function, the user must have access to Diagnose X'A0'. Upon successful completion of the function, condition code 0 will be set, and the specified register pair (Rx,Rx+1) will contain the 8-byte userid of the

CA Top Secret server.

If CA Top Secret has not been initialized since IPL of the CP system, then condition code 1 is set in response to the diagnose function and no registers are modified.

Authorization to execute Return CA Top Secret Server Userid Diagnose does not require ACTION(VMPRIV).

Chapter 5: Installation Exit

This section contains the following topics:

The VM Server Coding the Installation Exit

The Installation Exit gives a site the opportunity to alter the processing of key activities on the server machine. The Installation Exit program TSSINS is included as part of the server machine load deck during the server machine nucleus generation. Invoking the exit, once it has been installed, is controlled through the EXIT(ON|OFF) control option.

Control is passed to the Installation Exit at the following points:

- Directory post validation
- Message processing
- Minidisk post validation
- Password change
- Password phrase change
- Post-initiation
- Pre-initiation
- Resource access verification
- Resource post validation
- Security File change
- Session termination
- TSS command invocation
- Undefined user initiation
- Violation processing

In most cases, the exit program has the option of terminating the request, allowing the request to continue as is, altering the request, or bypassing security for the request.

Resource Access Exits

All exit points related to resource access checking (for example dataset and volume) are not driven unless the RDT entry for the resource class being validated has ATTR(EXIT) in the RDT table entry.

Standards and Format

Linkage into the exit follows standard O/S coding conventions. On entry, R13 contains the address of an 80-byte register save area, R14 contains the return address, R15 contains the entry point address, and R1 contains the address of the Installation Exit parameter list (#INSTXPL). While there are some fields which are constant across calls, the format of the parameter list and return codes to be used by the exit are dependent upon the function. The return code from the exit must be placed in R15 before returning. The Installation Exit must be written using re-entrant coding techniques. As a reminder, the server machine runs in its own operating system and does not use CMS. Therefore, no CMS functions (SVC 202, called routines, **etc.) are available.** Under no circumstances should the Installation Exit, including the Application Interface function, issue any form of the security diagnose (X'A0') with the exception of subcode zero (0). IUCV recursion and server “hang” will result. A list of system macros that can be used are given in the Appendix “TSSVM MACLIB.” These macros can be found in the TSSVM MACLIB included on the installation tape.

The following is a list of parameters that are static across all exit functions. Each field in the #INSTXPL (with the exception of TXAWORK) is a full word containing the address of the field.

TXAWORK

This is a workarea, 15 doublewords in size, which may be used for any purpose by the exit.

TXA#DWRD

The address of a static user doubleword. The address in this field is constant across all exit calls. You may wish to use this field as an anchor point for dynamic tables.

TXAFUNC

The function for which the exit has been invoked.

TXA#ACID

The ACIDNAME of the accessor in control at the time of the exit invocation.

TXA#USER

The name of the virtual machine under which the session was initiated.

TXA#TERM

The terminal ID, BATCH, AUTOLOG, or DISC for the user.

TXA#FACM

The systems facility matrix entry for the user. A mapping of this field may be found in the #FACMATX macro in TSSVM MACLIB.

TXA#MODE

The mode of the user. For pre-initiation and undefined user calls, this byte contains the facility mode. For all other calls, the byte contains the mode of the user.

TXA#FLAG

Flags which may be set by the Installation Exit follow:

- **TXAI$MC**-Mode (TXAIMODE) changed for the user. The mode remains as set by the exit for the duration of the session.
- **TXAI$AUD**-Begin auditing the user's activity.
- **TXAI$SUS**-Suspend the ACID

TXA#LANG

The one byte language preference code associated with the ACID or attached profile.

TXA#DRC

The most recent detail error reason code.

The following is a list of fields unique to pre-initiation, post-initiation, undefined ACID validation, password change, and password phrase change:

TXA#PASS

The old and new (if provided) password for the ACID.

TXAIPHRA

@ Old password phrase followed by new password phrase in 256 character areas.

- **+0** = 1 byte length (0 implies no old password phrase)
- **+1** = old password phrase data
- **+256** = 1 byte length (0 implies no new password phrase data)
- **+257** = new password phrase data

TXAIGRP

@ Group assigned by job or sign on process prior to actual initiation of sign on.

TXA#AISS

For autolog, the issuer of the AUTOLOG command. Additionally, the following settings exist for TXA#FLAG for pre and post-initiation calls:

- **$TXAIAUT**-The initiation request is for autolog.
- **$TXAIAAA**-An alternate ACID (ACID=) was entered on the command line with the LOGON or AUTOLOG command.
- **$TXAIFRC**-This is a forced logon, the virtual machine initiated before CA Top Secret.
- **$TXAIRND**-Indicates that the user has requested that a RANDOM new password be generated.

The following is a list of fields unique to minidisk, O/S data set, O/S volume, SFSDIR, general resource, post validation exits and ACTION(EXIT):

TXA#MDSK

The name of the minidisk for which a link is being attempted. The format is USERID.CUU for the minidisk name.

TXA#RESN

The name of the resource to which the ACID is trying to obtain access.

TXA#VMUS

The target VM userid for a CP command or diagnose.

TXA#DSN

The name of the O/S data set the ACID is trying to access.

TXA#VOL

The volume on which the data set in TXA#DSN may be found.

TXA#ACC

The requested access to the resource. For resource types which are included with CA Top Secret, these access levels are documented in the #AFLAGS macro. For site defined resources, a TSS LIS(RDT) RESCL(xxxxxxxx) shows the settings for resource access levels.

TXA#SFSD

SFS Directory Name Address.

TXA#SFSP

SFS Filepool Name Address.

TXA#SFSF

SFS CMS File Name Address.

TXA#RTYP

The type of resource which is being accessed. The resource types are documented in the #AFLAGS macro or may be listed by a TSS LIS(RDT) RESCL(xxxxxxxx).

TXARTYP2

@ 2-Byte Resource type (rescode from RDT). This parameter only exists when TXA#RTYP points to a x'FF' value.

The following is a list of fields unique to message and violation processing:

TXA#FLOG

The violation fast logging buffer. A mapping of this buffer may be found in the #FLOG macro in TSSVM MACLIB.

TXA#MBUF

A copy of the message to be issued. Each message contains the following format:

+0(1)-Real length of message text

+1(3)-Reserved

+4(x)-Variable message text

TXA#DRCE

The detail error reason code element mapping. The field is set by the DRC control option. A mapping of this field may be found in the #DRCD macro in TSSVM MACLIB.

TXA#MSGE

The message attribute element mapping. The field is set by the MSG control option. A mapping of this field can be found in the #MSGE macro in TSSVM MACLIB and can be listed by a TSS LIS(RDT) RESCL(xxxxxxxx).

The following is a list of fields unique to change processing:

TXA#RBUF

The change buffer. This field is valid only for TSS commands, when it points to the text of the TSS command issued. It should not be referenced for any other types of security file changes. To determine the type of security file change, check the CHGPTYPE field in the change parameter list (see TXA#CPL below).

TXA#CPL

The change parameter list. A mapping of this parameter list may be found in the #CHGPL macro in TSSVM MACLIB.

Appendix A: TSSVM MACLIB

The following is a description of system macros that can be found in the TSSVM MACLIB on the installation tape:

$$CPFTGT

Contains the CPF outbound command target machine list.

#AFLAGS

Contain equate characters for CA Top Secret defined resource types as well as attribute flags.

#AIPL

Contains the Application Interface parameter list passed with diagnose X'00A0'.

#CHGPL

Contains the change parameter list which includes such fields as the type of change, issuer, and subject of change.

#DFLAGS

Contains equate characters for all CA Top Secret Detail Error Reason Codes (DRCs).

#FACMATX

Maps a facility entry as defined by the FAC control option.

#FLOG

Contains the fast logging buffer user in violation recording.

#FREE

Contains the macro used for obtaining free storage.

#FRET

Contains the macro used for releasing free storage.

#INREG

Inserts a value into specified register.

#INSTXPL

Contains the Installation Exit parameter list passed to TSSINS.

#MSGE

Maps a message element entry as established by the MSG control option.

#SMF80

Audit information record layout.

TSS0100A

Zap template for changing the TSS0100A message.

TSS0101A

Zap template for changing the TSS0101A message.

TSS0102A

Zap template for changing the TSS0102A message.

TSS0115E

Zap template for changing the TSS0115E message.

TSS0120A

Zap template for changing the TSS0120A message.

Additionally there are five message members in TSSVM MACLIB. These can be used by the client to change the wording associated with the
CA Top Secret messages using each number.