CA Top Secret® for z/OS

Command Functions Guide

r15

Ninth Edition

This Documentation, which includes embedded help systems and electronically distributed materials (hereinafter referred to as the “Documentation”), is for your informational purposes only and is subject to change or withdrawal by CA at any time.

This Documentation may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA. This Documentation is confidential and proprietary information of CA and may not be disclosed by you or used for any purpose other than as may be permitted in (i) a separate agreement between you and CA governing your use of the CA software to which the Documentation relates; or (ii) a separate confidentiality agreement between you and CA.

Notwithstanding the foregoing, if you are a licensed user of the software product(s) addressed in the Documentation, you may print or otherwise make available a reasonable number of copies of the Documentation for internal use by you and your employees in connection with that software, provided that all CA copyright notices and legends are affixed to each reproduced copy.

The right to print or otherwise make available copies of the Documentation is limited to the period during which the applicable license for such software remains in full force and effect. Should the license terminate for any reason, it is your responsibility to certify in writing to CA that all copies and partial copies of the Documentation have been returned to CA or destroyed.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION “AS IS” WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, LOST INVESTMENT, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The use of any software product referenced in the Documentation is governed by the applicable license agreement and such license agreement is not modified in any way by the terms of this notice.

The manufacturer of this Documentation is CA.

Provided with “Restricted Rights.” Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

Copyright © 2014 CA. All rights reserved. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

CA Technologies Product References

This documentation set references the following CA products:

- CA ACF2™ for z/OS (CA ACF2)
- CA Common Services for z/OS (CA Common Services)
- CA Distributed Security Integration Server for z/OS (CA DSI Server)
- CA LDAP Server for z/OS (CA LDAP Server)
- CA Top Secret® for z/OS (CA Top Secret)

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.

Documentation Changes

The following documentation updates have been made since the last release of this documentation:

- PASSWORD Keyword (for Users)—Change a Password (see page 381)—Added descriptions for password syntax variables; clarified text about how non-administrative ACID types can work with the keyword.
- PASSWORD Keyword (for Administrators)—Assign a Password (see page 383)—Noted that, for z/OS, the administrator may set control option PWADMIN(YES) to enforce NEWPW and expiration interval restrictions on administrators; overhauled syntax descriptions; made miscellaneous corrections to the syntax.
- PKISERV Resource Class—Control Access to R_Pkiserv Administrative Functions (see page 740)—Added this section for a resource class that allows very granular control over access to R_PKIServ administrative functions.
- SYSAUTO Resource Class—IBM Automation control for z/OS resources (see page 788)—Added this section for a resource class that controls access to z/OS resources for IBM Automation Control.

Contents

Chapter 1: Introduction 31

Using Command Functions....31
Command Syntax....31
Entry Methods....32
Example: command syntax....33
Access Administration Panels....34
Online Processing of Commands....34
Command Response Messages....35
Administrative Authority....35
Generic Prefixing....35

Chapter 2: Command Functions 37

ADDTO Function—Add Resource Ownership and Attributes....37
Resource Ownership....38
Assign Attributes....39
ADDTO Applicable Keywords....40
ADMIN Function—Grant Administrative Authority....41
Examples: ADMIN function....44
CHKCERT Function—Check Certificates....44
COMPARE Function—Compare ACIDs....45
CREATE Function—Define a New ACID....46
Use of DEPARTMENT, DIVISION, and ZONE....47
CREATE Applicable Keywords....48
DEADMIN Function—Remove Administrative Authority....49
DELETE Function—Delete an ACID....49
EXPORT Function—Write a Certificate to a Data Set....51
GENCERT Function—Generate a Certificate....53
GENCERT Applicable Keywords....54
GENREQ Function—Generate a Certificate Request....55
HELP Function—Request Information....56
Example: HELP function....56
LIST Function—Display ACID Security Data....56
Hard Copy Listings....61
ACID Types....61
Profile Sorting....61
ACID Lists....62

LIST in a Shared Environment 62
LOCK Function—Lock Online Terminal 63
MODEL Function—Create a New User from a Model 63
Fields That Are Not Copied During User Modeling 66
MODIFY Function—Display or Alter Control Options 67
MOVE Function—Move an ACID 69
Example: MOVE function 70
Effects of MOVE if the TYPE Keyword is Omitted 70
Effects of MOVE Using the TYPE Keyword 71
P11TOKEN Function—Manage Certificates, Public Keys, and Private Key Objects 72
PERMIT Function—Permit Access to Resources 73
Multiple PERMITs 74
PERMIT Applicable Keywords 74
Duplicate Permissions 75
Selectively Revoking PERMITs 76
Using the GENERIC - NONGENERIC Attribute 77
Mask and Prefix for Other Resources 78
Data Set Prefixing 79
Minidisk Prefixing 82
REFRESH Function—Renew ACIDs 83
Example: REFRESH function 84
REKEY Function—Create Certificate from Existing Certificate 84
REMOVE Function—Remove Resource Ownership or Resources 89
RENAME Function—Change ID Assigned to ACID 91
Example: RENAME function 91
Global Records 92
RENEW Function—Renew a Digital Certificate 92
REPLACE Function—Change ACIDs Attributes 96
REPLACE Applicable Keywords 97
Example: REPLACE function 97
REVOKE Function—Revoke Resource Access 97
REVOKE Applicable Keywords 98
ROLLOVER Function—Replace Original Certificate 98
UNLOCK Function—Unlock Online Terminal 100
WHOAMI Function—Display ACID's Environment 101
WHOHAS Function—Display ACID's Resource Access 102
WHOHAS Applicable Keywords 103
Resource Access Information 104
Facility, Attribute, and Data Field Access Information 105
Administrative Authority Information 105
Example: WHOHAS function 106
WHOOWNS Function—Display Resource Owners 106

WHOOWNS Applicable Keywords 108
Example: WHOOWNS function 108

Chapter 3: Keywords 109

ACCESS Keyword—Control Access to Resources 109
Access Levels 109
Examples: ACCESS keyword 111
ACID Keyword—Authorize ACIDs 111
ACID Authority Levels 111
Examples: ACID keyword 113
ACIDPRFX Keyword—List ACIDs With Prefix 113
Examples: ACIDPRFX keyword 114
ACLST Keyword—Resource Access Level 114
Examples: ACLST keyword 116
ACTION Keyword—Assign Actions 117
Examples: ACTION keyword 119
AFTER and BEFORE Keywords—Specify Profile Order 121
Example: AFTER keyword 121
ALTNAME Keyword—SubjectAltname Values 122
Example: ALTNAME keyword 123
APPLDATA Keyword—Associate Data with PERMIT 123
Example: APPLDATA keyword 124
ARCHIVE Keyword—Archive a User's Security Record 124
Fields Not Copied during User Archive 126
ASSIZE Keyword—Maximum Address Space Size 126
Examples: ASSIZE keyword 127
ASUSPEND Keyword—Remove Suspension 127
Examples: ASUSPEND keyword 128
ASYMUSE Keyword—Asymmetric Key Usage Options 128
ATTR Keyword with the FDT—FDT Field Attributes 129
Example: ATTR Keyword 132
ATTR Keyword with the RDT—Resource Attributes 132
Attributes 132
Examples: ATTR keyword 135
AUDIT Keyword—ACID Activity 136
Examples: AUDIT keyword 137
BIND Keyword—Bind a Certificate to a PKCS#11 Token 137
BIND Authority Levels 138
Example: BIND keyword 139
BPECC Keyword—Generate Key with BPECC 139
CALENDAR Keyword—Create and Use a Calendar 140

Examples: CALENDAR keyword 141
CATEGORY Keyword—Category Specification 141
Examples: CATEGORY keyword 142
CERTFLAG Keyword—Set or Reset Digital Certificate Flags 142
CERTMAP Keyword—Certificate Filter Association 144
Examples: CERTMAP keyword 146
CERTNSER Keyword—Next Certificate Serial Number 148
Example: CERTNSER keyword 148
CHAIN Keyword—Display Certificate Information 148
CHKADDRS Keyword—Check Ticket Address 149
CNFAPP Keyword—Application Filter 150
Example: CNFAPP keyword 151
CNFUVAR Keyword—User Filter 151
Examples: CNFUVAR keyword 152
COMMAND Keyword—Specific Commands for ACID 152
Examples: COMMAND keyword 153
CONSOLE Keyword—Control Options Permissions 154
Examples: CONSOLE keyword 155
CPFNODE Keyword—Define CPF Node-Specific Options 156
CPFSYSID Keyword—Define CPF Global Options 157
CRITERIA Keyword—Additional Filter Criteria 159
Examples: CRITERIA keyword 161
CRITMAP Keyword—ACID to User Filter 161
Example: CRITMAP keyword 162
DATA Keyword with ADMIN—Authority to List Information 162
DATA Authority Levels 163
Rules 164
Examples: DATA keyword 165
DATA Keyword with WHOHAS—Interpret Resource Name 165
Examples: DATA keyword 166
DATA Keyword with LIST—Security Record Portion 168
Data Types 169
Examples: DATA keyword 172
DAYS Keyword with ACIDs—Restrict Access by Day 173
Examples: DAYS keyword 174
DAYS Keyword with SDT—Maintain a Calendar 174
Example: DAYS keyword 175
DCDSN Keyword—Specify a Data Set Containing a Certificate 175
DCENCRY Keyword—Encryption Key Value 177
Examples: DCENCRY keyword 177
DCLASS Keyword—Data Classification Class Definition 178
Examples: DCLASS Keyword 178

DEFACC Keyword—Resource Default Access 179
Access Levels 179
Examples: DEFACC keyword 180
DEFAULT Keyword—Key Ring Default Certificate 180
Example: DEFAULT keyword 181
DEFNODES Keyword—Default Remote Nodes 181
Examples: DEFNODES keyword 182
DEFTKTLF Keyword—Default Ticket Life 183
Example: DEFTKTLF keyword 184
DEPARTMENT Keyword—Department ACID Assignment 184
Examples: DEPARTMENT keyword 185
DESCRIPT Keyword—List Description 186
Example: DESCRIPT keyword 187
DFLTGRP Keyword—ACID Default Group 187
Examples: DFLTGRP keyword 187
DFLTSLBL Keyword—Default SECLABEL Association 188
Example: DFLTSLBL keyword 188
DIGICERT Keyword—Identify a Digital Certificate 188
DISPLAY Keyword—FDT Display Field 191
Examples: DISPLAY Keyword 191
DIVISION Keyword—Division Data 192
Examples: DIVISION keyword 192
DSA—Generate Keys with DSA 193
Examples: DSA keyword 193
DUFUPD Keyword—INSTDATA and Security Record Updates 194
Examples: DUFUPD keyword 194
DUFXTR Keyword—INSTDATA and Security File Data Extraction 195
Examples: DUFXTR keyword 195
EIMDOMAIN Keyword—EIM Distinguished Name 195
Example: EIMDOMAIN keyword 196
EIMLOCREG Keyword—EIM Local Registry Name 197
Example: EIMLOCREG keyword 197
EIMOPTION Keyword—EIM Connection Permission 198
EIMPROF Keyword—Supply an EIM Profile Name for an ACID 198
ENCRYPT Keyword—Encryption Level 201
Examples: ENCRYPT keyword 203
ENCRYPT Keyword—Encryption Level Override 203
EXCLUDE Keyword—Date Exclusion 205
Example: EXCLUDE keyword 205
EXPIRES Keyword—Expiration Removal 205
Example: EXPIRES keyword 206
FACILITY Keyword—Specify Facilities 206

Examples: FACILITY keyword 207
FAILLOAD Keyword—SIGVER record program definition 209
Examples: FAILLOAD keyword 210
FDTCODE Keyword—Field Code Data 210
Example: FDTCODE keyword 210
FDTNAME Keyword—FDT Record Field Names 211
Examples: FDTNAME Keyword 212
FIRST Keyword—Add a Profile to the Beginning of a List 212
Example: FIRST keyword 213
FOR Keyword—Specify ACID's or Permission's Life 213
FOR Related Keywords 213
Expired Access 214
Examples: FOR keyword 214
FORCE Keyword—Bypass Checks 215
FORMAT Keyword—Specify Certificate Format 216
Examples: FORMAT keyword 217
FROMICSF Keyword—Specify Certificate Label 218
Examples: FROMICSF keyword 218
GAP Keyword—Specify if a Profile is Globally Administered 219
Examples: GAP keyword 219
GID Keyword—Assign a Group ID for USS Group Security 219
GROUP Keyword—Define Groups to an ACID 222
Examples: GROUP keyword 222
HOME Keyword (MODLUSER)—Define a Prototype HOME Subdirectory 223
HOME Keyword (Session User)—Define USS Entry Subdirectory 225
ICSF Keyword—Place Private Key in ICSF 227
Examples: ICSF keyword 228
IDLABEL Keyword—Specify Identity Propagation Label 228
Example: IDLABEL keyword 229
IDMAPDN, IDLABEL, and IDMAPRN Usage Rules 229
IDMAP Keyword—z/OS Identity Propagation Mapping 229
Example: IDMAP Keyword 230
Activating IDMAP Profile Records 230
IDMAPDN Keyword—Specify Identity Distinguished Name 231
Example: IDMAPDN keyword 232
Using Filters for the Distinguished Name and Registry Name 232
IDMAPRN Keyword—Specify Identity Registry Name 233
IDNFILTR Keyword—Specify a Distinguished Name Filter 235
Example: IDNFILTR keyword 236
IESFL1 Keyword—Assign Attributes Interactive User Interface 237
Examples: IESFL1 keyword 238
IESFL2 Keyword—Assign Interactive Interface User Attributes 238

Examples: IESFL2 keyword 239
IESINIT Keyword—Assign Interactive Interface User Name 239
Examples: IESINIT keyword 240
IESSYNM Keyword—Interactive User Interface Synonyms Model 240
Examples: IESSYNM keyword 241
IESTYPE Keyword—Assign User Type to IUI 241
Examples: IESTYPE keyword 242
IESVCAT Keyword—Assign Default VSAM User Catalog 242
Examples: IESVCAT keyword 242
IMPORT Keyword—Import Certificate from PKCS#11 Token 243
IMSMSC Keyword—Determine MSC Security Level 244
Examples: IMSMSC keyword 245
INCLUDE Keyword—Include Dates 246
Example: INCLUDE keyword 246
INSTDATA Keyword—Record ACID Information 246
Examples: INSTDATA keyword 247
INTO Keyword—Store TSS Commands to a PDS Data Set 248
ISSUERDN and SERIALNUM Keywords—Identify Certificate 250
Examples: ISSUERDN and SERIALNUM keyword 251
JOBNAME Keyword—Bring Data Set into Hyperspace 251
Examples: JOBNAME keyword 252
KERBLINK Keyword—Define Foreign Kerberos Users 252
Examples: KERBLINK keyword 254
KERBNAME Keyword—Specify Local Principals as Users 254
Examples: KERBNAME keyword 255
KERBPASS Keyword—Foreign Password 256
Example: KERBPASS keyword 257
KERBPASS Keyword—REALM Record Password 257
Examples: KERBPASS keyword 259
KERBSEGM Keyword—Kerberos Principal User Record 259
Example: KERBSEGM keyword 260
KERBUSER Keyword—Map Foreign Principal Names 260
Example: KERBUSER keyword 260
KERBVIO Keyword—Count Attempts to Use a Key 261
Example: KERBVIO keyword 261
KEYRING Keyword—Add a Key Ring to a User 262
Examples: KEYRING keyword 263
KEYSIZE Keyword—Specify Key Size 264
Example: KEYSIZE keyword 265
KEYSMSTR Keyword—Define Password Key Name 265
Examples: KEYSMSTR keyword 266
KEYUSAGE—Specify Key Usage Extension 267

LABLCERT Keyword—Specify Certificate Label 268
Example: LABLCERT keyword 269
LABLCMAP Keyword—Specify Certificate Filter Label 269
Example: LABLCMAP keyword 270
LABLPKDS Keyword—Specify Certificate Label 270
Examples: LABLPKDS keyword 272
LABLRING Keyword—Key Ring Label 272
Examples: LABLRING keyword 273
LANGUAGE Keyword—Specify Language Preference 273
Examples: LANGUAGE keyword 274
LDAPDEST Keyword—Define Node to LDAP Node List 274
LDAPNODE Keyword—Define LDAP Nodes 275
LDS Keyword—Add or Remove LDS Attribute 283
Examples: LDS keyword 284
LDSYSID Keyword—Define LDS Global Options 284
LIBRARY Keyword—Specify Privileged Program Library 285
Example: LIBRARY keyword 286
LINKID Keyword—Identify LUs for APPC Conversation 287
Examples: LINKID keyword 288
LINKNAME Keyword—Specify Foreign Realm 289
Example: LINKNAME keyword 289
LINUXNAM Keyword—Define Linux User Name 290
Example: LINUXNAM keyword 290
LINUXNODE Keyword—Define LINUX Nodes 290
Example: LINUXNODE 291
LNXENTS Keyword—Provide Sign on Information 291
Examples: LNXENTS keyword 293
LOADLIB Keyword—Specify SIGVER record program library definition 293
Examples: LOADLIB keyword 294
LTIME Keyword—Minutes Until Terminal Locks 295
Examples: LTIME keyword 295
MAPDATA Keyword—Define MAPDATA Field 296
Examples: MAPDATA keyword 297
MAPREC Keyword—Reference MAP Record 298
Examples: MAPREC keyword 299
MASKDATA Keyword—SDT Record Mask 300
Examples: MASKDATA keyword 302
MASTFAC (z/OS and z/VSE) Keyword—Override Default Facility 303
Examples: MASTFAC keyword 303
MASTFAC (z/VM) Keyword—Associate Virtual Machine to a Facility 304
Examples: MASTFAC keyword 304
MASKREC Keyword—MASK Record Name 305

Example: MASKREC keyword 306
MASKREC Keyword—Overlay FCT Values 307
Examples: MASKREC keyword 308
MATCHLIM Keyword—Limit Audit Activity 309
MAXLEN Keyword—Define FDT Maximum Length 311
Example: MAXLEN Keyword 311
MAXLEN Keyword—Define Maximum Permission Length 312
Example: MAXLEN keyword 312
MAXTKTLF Keyword—Specify Maximum Ticket Life 313
Examples: MAXTKTLF keyword 314
MCSALTG Keyword—Alternate Recovery Group 314
Examples: MCSALTG keyword 315
MCSAUTH Keyword—Authorize Commands 316
Examples: MCSAUTH keyword 317
MCSAUTO Keyword—Assign AUTO Keyword 317
Examples: MCSAUTO keyword 317
MCSCMDS Keyword—Specify System 318
Example: MCSCMDS keyword 318
MCSDOM Keyword—Specify Delete Messages Received 319
Examples: MCSDOM keyword 319
MCSHC Keyword—Receive Hardcopy Message Set 320
Example: MCSHC keyword 320
MCSINTIDS Keyword—Receive Messages for Console ID=0 321
Example: MCSINTIDS keyword 321
MCSKEY Keyword—Assign Key Keyword 322
Examples: MCSKEY keyword 322
MCSLEVL Keyword—Specify Messages Received 323
Example: MCSLEVL keyword 324
MCSLOGC Keyword—Specify Hard Copy Log 324
Example: MCLOGC keyword 324
MCSMFRM Keyword—Console Message Format 325
Examples: MCSMFRM keyword 326
MCSMGID Keyword—Migration ID 326
Examples: MCSMGID keyword 326
MCSMON Keyword—System Event Monitoring 327
Example: MCSMON keyword 328
MCSROUT Keyword—Console Routing Codes 328
Example: MCSROUT keyword 328
MCSSTOR Keyword—Message Queue Storage 329
Example: MCSSTOR keyword 329
MCSUD Keyword—Specify Messages Received 330
Example: MCSUD keyword 330

MCSUNKNIDS Keyword—Receive Messages for Unknown Console IDs 331
Example: MCSUNKNIDS keyword 331
MEMLIMIT Keyword—Maximum Memory Space 332
Examples: MEMLIMIT keyword 333
MINTKTLF Keyword—Minimum Ticket Life 333
Example: MINTKTLF keyword 334
MISC1 Keyword—Authority to Administer Functions 334
MISC1 Authority Level 335
Examples: MISC1 keyword 336
MISC2 Keyword—Authority to Administer Functions 336
MISC2 Authority Levels 337
Examples: MISC2 keyword 338
MISC3 Keyword—Authority to Administer Functions 338
MISC3 Authority Levels 338
Examples: MISC3 keyword 339
MISC4 Keyword—Authority to Administer Functions 339
MISC4 Authority Levels 340
Examples: MISC4 keyword 340
MISC5 Keyword—Authority to Administer Functions 341
MISC5 Authority Levels 341
Examples: MISC5 keyword 342
MISC7 Keyword—Authority to Administer Functions 342
MISC7 Authority Levels 342
MISC8 Keyword—Authority to Administer Functions 343
MISC8 Authority Levels 344
Examples: MISC8 keyword 345
MISC9 Keyword—Authority to Administer Functions 345
MISC9 Authority Levels 346
Examples: MISC9 keyword 347
MMAPAREA Keyword—Maximum Data Space Pages for HFS Mappings 347
Examples: MMAPAREA keyword 348
MODE Keyword—Operating Mode 348
MODE Keyword—Specify Operating MODE 349
Examples: MODE keyword 349
MRO Keyword—Maintain Security Records 349
Examples: MRO keyword 350
MULTIPW Keyword—Maintain Multiple Password 350
Examples: MULTIPW keyword 351
NADATE and NATIME Keywords—Date Certificate Expires 352
Example: NADATE and NATIME keywords 353
NAME Keyword—Associate ACID with a Name 353
Example: NAME keyword 354

NBDATE and NBTIME Keywords—Date Certificate Activates 354
Example: NBDATE and NBTIME keywords 355
NEWDIGIC Keyword—Name Certificate 355
Example: DIGICERT keyword 356
NEWLABLC Keyword—Label Associated with Certificate 356
Example: NEWLABLC keyword 357
NISTECC Keyword—Generate Key with NISTECC 357
NOADSP Keyword—Prevent Data Set Security 358
Examples: NOADSP keyword 358
NOATS Keyword—Fail Automatic Terminal Signons 359
Examples: NOATS keyword 359
NODSNCHK Keyword—Bypasses Access Security Checks 360
Examples: NODSNCHK keyword 360
NOLCFCHK Keyword—Execute any Command 361
Examples: NOLCFCHK keyword 361
NOPERMIT Keyword—Prevent Automatic Access 362
Examples: NOPERMIT keyword 362
NOPWCHG Keyword—Prevent Password Changes 363
Examples: NOPWCHG keyword 363
NOREFRESH Keyword—Prevent Logons Copying Profile 364
NORESCHK Keyword—Bypass Security Checking 364
Examples: NORESCHK keyword 365
NOSUBCHK Keyword—Bypass Alternate Security Checking 365
Examples: NOSUBCHK keyword 366
NOSUSPEND Keyword—Bypass Violation Suspensions 366
Examples: NOSUSPEND keyword 366
NOVMDCHK Keyword—Bypass Minidisk Link Checking 367
Examples: NOVMDCHK keyword 367
NOVOLCHK Keyword—Bypass Volume Level Checking 368
Examples: NOVOLCHK keyword 368
OECPUTM Keyword—Maximum CPUTIME for a Dubbed Process 369
Examples: OECPUTM keyword 369
OEFILEP Keyword—Maximum Files per Process 370
Examples: OEFILEP keyword 370
OIDCARD Keyword—Prompt for Identity Card 371
Examples: OIDCARD keyword 371
OMVSPGM Keyword—Add or Remove a Program 372
Examples: OMVSPGM keyword 372
ONAME1 Keyword—Data Classification Owner Data 373
Example: ONAME1 Keyword 373
ONAME2 Keyword—Data Classification Owner Data 374
Example: ONAME2 Keyword 374

OPCLASS Keyword—Maintain CICS Operator Classes 375
Examples: OPCLASS keyword 375
OPIDENT Keyword—Maintain CICS Operator Identification 376
Examples: OPIDENT keyword 376
OPPRTY Keyword—Maintain CICS Operator Priority 377
Examples: OPPRTY keyword 377
OPTIME keyword—Maintain CICS Timeout 378
Examples: OPTIME keyword 378
OWNER1 Keyword—Data Classification Owner Definition 379
Examples: OWNER1 Keyword 379
OWNER2 Keyword 380
Examples: OWNER2 keyword 381
PASSWORD Keyword (for Users)—Change a Password 381
PASSWORD Keyword (for Administrators)—Assign a Password 383
PCICC Keyword—Generate Keys with PCI Cryptographic Coprocessor 386
Examples: PCICC keyword 387
PHRASE Keyword—Assign Password Phrase 388
PHYSKEY Keyword—Support External Authentication Devices 389
Examples: PHYSKEY keyword 390
PKCSPASS Keyword—Specify Certificate Password 390
Example: PKCSPASS 391
POSIT Keyword—Class POSIT Number 391
Example: POSIT keyword 391
PREFIX Keyword—Record Matching by Partial Key 392
Examples: PREFIX keyword 392
PRIVPGM Keyword—Specify Programs in Control 392
PROCNAME Keyword—Define STC Procedure 395
Examples: PROCNAME keyword 397
PROCUSER Keyword—Maximum Number of Processes 399
Examples: PROCUSER keyword 399
PROFILE Keyword—Maintain Profiles 400
Examples: PROFILE keyword 401
PROGRAM Keyword—Specify SIGVER record program definition 401
Examples: PROGRAM keyword 402
PRXBINDDN Keyword—LDAP Server Distinguished Name 403
PRXBINDPW Keyword—Authenticate LDAP Server 404
PRXKRBREG Keyword—Kerb Registry Name 405
Example: PRXKRBREG keyword 405
PRXLDAPHST Keyword—LDAP Server URL and Port 406
PRX509REG Keyword—Kerb Registry Name 407
PSTKAPPL Keyword—Application ID 408
Example: PSTKAPPL keyword 408

PSUSPEND Keyword—Prevent Access after PTHRESH Violation 409
Examples: PSUSPEND keyword 409
PSWDPHR Keyword—Test Password Phrase 410
Example: PSWDPHR keyword 410
RANGE Keyword—Time Interval 411
Example: RANGE keyword 411
RCLASS Keyword—Data Classification Resource Class Definition 412
Examples: RCLASS Keyword 412
REALM Keyword—Define Realms 413
Examples: REALM keyword 414
REALM(FOREIGN_REALM) Keyword—Foreign Node Label 415
Example: REALM keyword 415
REALM(KERBDFLT) Keyword—Define Local Realm 416
Example: REALM(KERBDFLT) Keyword 416
REALMNAME Keyword—Secureway Realm 417
REALMNAME(Local Realm) Keyword—Local Realm Name 418
Example: REALMNAME keyword 419
REALMNAME Keyword—Specify Local and Foreign Realm Name 420
Example: REALMNAME keyword 421
RECDATA Keyword—RECORD Field Characteristics 422
Examples: RECDATA keyword 423
RECORD Keyword—RLP Record Name 424
Example: RECORD Keyword 424
RESCLASS Keyword—Resource Classes 425
Examples: RESCLASS keyword 426
resclass(resource) Keyword—Associate Resource with SECLABEL 427
Example: resource keyword 427
RESCODE Keyword—Resource Class Abbreviation 428
Examples: RESCODE keyword 429
RESNAME Keyword—Data Classification Resource Entity 429
Examples: RESNAME Keyword 430
RESOURCE Keyword—Global Resource Class Administration Authority 431
RESOURCE Authority Levels 432
Access Levels 432
Examples: RESOURCE keyword 434
RESOURCE Keyword—Individual Resource Class Administration Authority 434
RESOURCE Authority Levels 435
Access Levels 435
Example: resource Keyword 436
RESOWNER Keyword—Maintain SMS ACID 437
Examples: RESOWNER keyword 437
RETAIN Keyword—Leave Data Set in Hyperspace 438

Examples: RETAIN keyword 438
RINGDATA Keyword—Add Certificate to a Ring 439
Examples: RINGDATA keyword 440
RSTDACC keyword—Control Access to Resources 440
Example: RSTDACC keyword 441
SCOPE Keyword—Allow Authority 442
Examples: SCOPE keyword 443
SCTYKEY Keyword—Specify CICS Security Keys Use 444
Examples: SCTYKEY keyword 444
SDNFILTR Keyword—Portion of DN for Filter 445
Example: SDNFILTR keyword 446
SDTFNAME Keyword—Remove SDT Record Field 447
Example: REMOVE keyword 447
SECLABEL Keyword—Security Labels 448
Examples: SECLABEL keyword 449
SECLEVEL Keyword—Security Level 449
SEGMENT Keyword—Field Segments 451
Examples: SEGMENT Keyword 452
SELDATA Keyword—SDT SELDATA Field 453
Example: SELDATA keyword 454
Parentheses with SELDATA 455
SELECT Keyword—Record Control 456
Examples: SELECT keyword 458
SERIALNUM Keyword—Certificate Serial Number 459
Example: SERIALNUM keyword 459
SESSKEY Keyword—Session Keys 460
Examples: SESSKEY keyword 461
SHMEMMAX Keyword—Maximum Shared Memory 462
Examples: SHMEMMAX keyword 463
SIGAUDIT Keyword—SIGVER record program definition 463
Examples: SIGAUDIT keyword 464
SIGNALG—Certificate Signing Algorithm 464
SIGNMULTI Keyword—Allow Multiple Sign Ons 465
Examples: SIGNMULTI keyword 466
SIGNWITH Keyword—Specify a Private Key to Sign a New Certificate 466
SIGREQ Keyword—SIGVER record program definition 467
Examples: SIGREQ keyword 468
SITRAN Keyword—CICS Automatic Transaction 468
Examples: SITRAN keyword 469
SMASYS and SMANODE Keywords—Startup, Recovery, and Thread Options 470
Example: SMASYS keyword 471
SMSAPPL Keyword—Maintain Default SMS Application 472

Examples: SMSAPPL keyword 472
SMSDATA Keyword—Default SMS Data Class 473
Examples: SMSDATA keyword 473
SMSMGMT Keyword—Default SMS Management Class 474
Examples: SMSMGMT keyword 474
SMSSTOR Keyword—Default SMS Storage Class 475
Examples: SMSSTOR keyword 475
SNAME Keyword—Map User from Lotus Notes 476
Examples: SNAME keyword 476
SOURCE Keyword—Source Prefixes 477
Examples: SOURCE keyword 477
START Keyword—Activation Date 478
Example: START keyword 478
STCACT Keyword—Operator Accountability 479
Examples: STCACT keyword 480
SUBJECTN Keyword—Certificate DN 481
Example: SUBJECTN keyword 482
SUSPEND Keyword—Prevent Access after Violation 482
SYMXCERT Keyword—Symmetric Certificate Labels 484
Example: SYMXCERT keyword 484
SYMXPORT Keyword—Symmetric Key Export Options 485
Example: SYMXPORT keyword 485
SYMXKEYS Keyword—Symmetric PKDS Labels 486
Example: SYMXKEYS keyword 486
SYSID Keyword—Identify System 486
TARGET Keyword—Node Assignment 488
Examples: TARGET keyword 490
THREADS Keyword—Maximum Number of PTHREAD Created Threads 491
Examples: THREADS keyword 491
TIMEREC Keyword—Time Range Label 492
TIMEREC Access 493
Example: TIMEREC keyword 493
TIMES Keyword—Access Hours 494
Time Ranges 495
Examples: TIMES keyword 495
TOKENADD Keyword—Create PKCS#11 Token 496
Example: TOKENADD keyword 496
TOKENDEL Keyword—Delete PKCS#11 Token 497
Example: TOKENDEL keyword 497
TOKENLST Keyword—PKCS#11 Token Certificate Object Information 498
Example: TOKENLST keyword 498
TRACE Keyword—Diagnostic Trace Activation 499

Examples: TRACE keyword 499
TRANSACTIONS Keyword—Specific Transaction 500
Examples: TRANSACTIONS keyword 501
TRUST Keyword—Associate a Certificate to a User 502
Example: TRUST keyword 503
TSOCOMMAND Keyword—Default TSO Logon Command 504
Examples: TSOCOMMAND keyword 504
TSODEST Keyword—TSO Default Destination Identifier 505
Examples: TSODEST keyword 505
TSOHCLASS Keyword—TSO Default Hold Class 506
Examples: TSOHCLASS keyword 506
TSOJCLASS Keyword—TSO Default Job Class 507
Examples: TSOJCLASS keyword 507
TSOLPROC Keyword—TSO Default Proc 508
Examples: TSOLPROC keyword 508
TSOLSIZE Keyword—TSO Default Region Size 509
Examples: TSOLSIZE keyword 509
TSOMCLASS Keyword—TSO Default Message Class 510
Examples: TSOMCLASS keyword 510
TSOMPW Keyword—Support Multiple TSO UADS Passwords 511
Examples: TSOMPW keyword 511
TSOMSIZE Keyword—TSO Maximum Region Size 512
Examples: TSOMSIZE keyword 512
TSOOPT Keyword—Assign Default TSO Options 513
Examples: TSOOPT keyword 513
TSOSCLASS Keyword—TSO Default SYSOUT Class 514
Examples: TSOSCLASS keyword 514
TSOUDATA Keyword—TSO Data Field 515
Examples: TSOUDATA keyword 515
TSOUNIT Keyword—TSO Default Unit Name 516
Examples: TSOUNIT keyword 516
TYPE Keyword—ACID Type 517
Example: TYPE keyword 517
TZONE Keyword—Physical Time Zone 518
Examples: TZONE keyword 518
UID Keyword—Assign a User ID to a USS User 518
UNAME Keyword—Map User from Novell Directory Services 521
Examples: UNAME keyword 521
UNBIND Keyword—Remove Certificate from PKCS#11 Token 522
UNDERCUT Keyword—Transfer Resource Ownership 523
Examples: UNDERCUT keyword 524
UNTIL Keyword—Expire Date 524

Examples: UNTIL keyword 525
USAGE Keyword—Certificate Trust Level 526
Examples: USAGE keyword 527
USER Keyword—Maintain Resource Access 527
Examples: USER keyword 528
USERNL1 and USERNL2 Keywords—CTS National Language 528
USING Keyword—Specify a Model ACID 528
USING Keyword with COMPARE—Specify ACID to Compare 531
VMUSER Keyword—User Authority 532
Example: VMUSER keyword 532
VSECATBT Keyword—Catalog B-transients Rights 532
Examples: VSECATBT keyword 533
VSEMCON Keyword—Master Console Rights 533
Examples: VSEMCON keyword 533
VSERDD Keyword—Read Directory Rights 534
Examples: VSERDD keyword 534
VSESYSAD Keyword—Security Administrator Rights 534
Examples: VSESYSAD keyword 535
VSUSPEND Keyword—Remove ACID Suspension 535
Examples: VSUSPEND keyword 535
WAACCNT Keyword—APPC Processing Account Number 536
Examples: WAACCNT keyword 536
WAADDRn Keyword—Additional SYSOUT Information 537
Example: WAADDRn keyword 537
WABLDG Keyword—SYSOUT Building Delivery 538
Examples: WABLDG keyword 538
WADEPT Keyword—SYSOUT Department Delivery 539
Example: WADEPT keyword 539
WAIT Keyword—Synchronous or Asynchronous Processing 540
Examples: WAIT keyword 541
WANAME Keyword—SYSOUT Person Delivery 541
Example: WANAME keyword 542
WAROOM Keyword—SYSOUT Room Delivery 542
Example: WAROOM keyword 542
XCOMMAND Keyword—Command Restrictions 543
Examples: XCOMMAND keyword 544
XSUSPEND Keyword—Suspend ACID 544
Examples: XSUSPEND keyword 545
XTRANSACTIONS Keyword—ACID Transaction Restriction 545
Examples: XTRANSACTIONS keyword 546
YEAR Keyword—Calendar Year 546
Example: YEAR keyword 547

ZONE Keyword—Zone ACIDs....547
Example: ZONE keyword....548

Chapter 4: Resource Classes 549

About Resource Classes....553
ABSTRACT Resource Class—Secure Resource Classes....554
Examples: ABSTRACT resource class....555
ACID Resource Class—Secure ACIDS....556
Examples: ACID resource class....556
APPCLU Resource Class—Secure APPCLU Links....557
Examples: APPCLU resource class....558
APPCPORT Resource Class—Specify VTAM LU Name....559
Examples: APPCPORT resource class....560
APPCSI Resource Class—Secure APPC Side Information Files....561
Masking....562
Examples: APPCSI resource class....562
APPCTP Resource Class—Secure APPC Transaction Programs....563
Masking....564
Examples: APPCTP resource class....564
APPLICATION Resource Class—Secure IMS Application Group Names....565
Examples: APPLICATION resource class....566
AREA Resource Class—Secure CA-IDMS Database Areas....566
Examples: AREA resource class....567
CAADMIN Resource Class—Secure CAADMIN Administration....568
Examples: CAADMIN resource class....569
CACMGR Resource Class-Secure CACMGR....570
Examples: CACMGR resource class....571
CACCFDSN Resource Class—Secure CACCFDSN Data Sets....572
Examples: CACCFDSN resource class....573
CACCFMEM Resource Class—Secure CACCFMEM Members....574
Examples: CACCFMEM resource class....575
CACMD Resource Class—Secure CA Commands....576
Examples: CACMD resource class....577
CAGSVX Resource Class—Secure CAGSVX for SYSVIEW....578
Examples: CAGSVX resource class....579
CALIBMEM Resource Class—Secure CALIBMEM Members....580
Example: CALIBMEM resource class....581
CAREPORT Resource Class—Secure Reports....582
Masking....583
Example: CAREPORT resource class....583
CASECAUT Resource Class—Authorize Restricted Administrative Priviliges....584

CATAPE Resource Class—Secure Tape Volumes 586
Example: CATAPE resource class 587
CAVAPPL Resource Class—Secure CAVAPPL for CAVMAN 588
Examples: CAVAPPL resource class 589
CBIND Resource Class—Secure CBIND for z/OS Objects 590
Examples: CBIND resource class 591
CIMS Resource Class—Secure IMS Commands 592
Examples: CIMS resource class 593
CPCMD Resource Class—Secure CP Commands 594
Examples: CPCMD resource class 595
CPU Resource Class—Secure CPU Access 596
Examples: CPU resource class 597
CSFKEYS Resource Class—Secure ICSF CSFKEYS 598
Examples: CSFKEYS resource class 599
CSFSERV Resource Class—Secure ICSF CSFSERV 600
Examples: CSFSERV resource class 601
CTSDB2 Resource Class—Secure CTS DB2ENTRY and DB2TRAN Resources 602
DATABASE Resource Class—Secure Databases 603
Examples: DATABASE resource class 604
DB2 Resource Class—Secure DB2 Databases 605
Example: DB2 resource class 607
DB2BUFFP Resource Class—Secure DB2 Buffer Pools 608
Examples: DB2BUFFP resource class 609
DB2COLL Resource Class—Secure DB2 Collection ID 610
Examples: DB2COLL resource class 611
DB2DBASE Resource Class—Secure DB2 Databases 612
Examples: DB2DBASE resource class 613
DB2PKG Resource Class—Secure DB2 Packages 614
Examples: DB2PKG resource class 615
DB2PLAN Resource Class—Secure DB2 Plans 616
Examples: DB2PLAN resource class 617
DB2SEQ Resource Class—Identify DB2 Sequences 618
DB2STOGP Resource Class—Secure DB2 Storage Groups 620
Examples: DB2STOGP Resource Class 621
DB2SYS Resource Class—Secure DB2 Privileges and Authorities 622
Privileges 623
Examples: DB2SYS resource class 623
DB2TABLE Resource Class—Secure DB2 Tables 624
Examples: DB2TABLE Resource Class 625
DB2TABSP Resource Class—Secure DB2 Table Spaces 626
Examples: DB2TABSP resource class 627
DBD Resource Class—Secure IMS Database Name 628

Examples: DBD resource class 629
DCSS Resource Class—Secure Segments 630
Example: DCSS resource class 631
DCT Resource Class—Secure CICS DCT Data 632
Examples: DCT resource class 633
DCTABLE Resource Class—Secure CA SYSVIEW DCTABLE 634
Examples: DCTABLE resource class 635
DEVICES Resource Class—Secure Devices 636
Examples: DEVICES resource class 637
DFTABLE Resource Class—Secure CA SYSVIEW DFTABLE 638
Examples: DFTABLE resource class 639
DIAGNOSE Resource Class—Secure CP Diagnose Codes 640
Examples: DIAGNOSE resource class 641
DIRECTRY Resource Class—Secure DIRECTRY for TSSVM 642
Examples: DIRECTRY resource class 643
DLFCLASS Resource Class—Determine DLF Record Access 644
Example: DLFCLASS resource class 645
DRTABLE Resource Class—Secure CA SYSVIEW DRTABLE 646
Examples: DRTABLE resource class 647
DSNAME Resource Class—Secure Data Sets 648
Example: DSNAME resource class 649
DSPACE Resource Class—Restrict Access to Databases 650
Examples: DSPACE resource class 651
DSTABLE Resource Class—Secure CASYSVIEW DSTABLE 652
Examples: DSTABLE resource class 653
DTADMIN Resource Class—Secure CASYSVIEW DTADMIN 654
Examples: DTADMIN resource class 655
DTSYSTEM Resource Class—Secure CASYSVIEW DTSYSTEM 656
Examples: DTSYSTEM resource class 657
DTTABLE Resource Class—Secure CASYSVIEW DTTABLE 658
Examples: DTTABLE resource class 659
DTUTIL Resource Class—Secure CASYSVIEW DTUTIL 660
Examples: DTUTIL resource class 661
DXTABLE Resource Class—Secure CASYSVIEW DXTABLE 662
Example: DXTABLE resource class 663
EJBROLE Resource Class—Secure z/OS Enterprise Beans 664
FCT Resource Class—Secure CICS File Control Table Entries 666
Examples: FCT resource class 667
FIELD Resource Class—Secure Database Fields 668
Protection 669
Examples: FIELD resource class 669
FSACCESS Resource Class—Secure zFS File System Container 670

GEJBROLE Resource Class—Secure z/OS Enterprise Beans....672
GSOMDOBJ Resource Class—Secure z/OS GSOMDOBJ Objects....674
Examples: GSOMDOBJ resource class....675
HFSSEC Resource Class—Secure HFS Files....676
Examples: HFSSEC resource class....677
IBMFAC Resource Class—Determine IBM Facilities Ownership....678
Examples: IBMFAC resource class....679
IBMGROUP Resource Class—Assign Group Names....680
Examples: IBMGROUP resource class....681
IUCV Resource Class—Secure IUCV Target Users....682
Examples: IUCV resource class....683
JCT Resource Class—Secure JCT Entries....684
Example: JCT resource class....685
JESINPUT Resource Class—Secure JES Job Entry Sources....686
Masking....687
Examples: JESINPUT resource class....687
JESJOBS Resource Class—Secure Jobs....688
Masking....689
Examples: JESJOBS resource class....689
JESSPOOL Resource Class—Secure JES Spool Data Sets....690
Masking....691
Examples: JESSPOOL resource class....691
JOBNAME Resource Class—Secure Jobname....692
Examples: JOBNAME resource class....693
LDAP Resource Class—Secure LDAP Administration....694
LFSCLASS Resource Class—Secure MLF LFSCLASS....696
Examples: LFSCLASS resource class....697
LOGSTRM Resource Class—Secure System Logger....698
MGMTCLAS Resource Class—Secure Management Classes....699
Examples: MGMTCLAS resource class....700
MQADMIN Resource Class—Secure MQM Commands....701
Masking....702
Examples: MQADMIN resource class....702
MQCMDS Resource Class—Secure MQM Commands....703
Masking....704
Examples: MQCMDS resource class....704
MQCONN Resource Class—Secure MQM Connections....705
Masking....706
Examples: MQCONN resource class....706
MQNLIST Resource Class—Secure MQM Name Lists....707
Masking....708
Examples: MQNLIST resource class....708

MQPROC Resource Class—Secure MQM Processes 709
Masking 710
Example: MQPROC resource class 710
MQQUEUE Resource Class—Secure the MQM Queue 711
Masking 712
Examples: MQQUEUE resource class 712
MXADMIN Resource Class—Secure MXM Commands 713
Masking 714
MXNLIST Resource Class—Secure MXM Name Lists 715
Masking 716
Examples: MXNLIST resource class 716
MXPROC Resource Class—Secure MXM Topics 717
Masking 718
Example: MXPROC resource class 718
MXQUEUE Resource Class—Secure the MXM Queue 719
Masking 720
Examples: MXQUEUE resource class 720
MXTOPIC Resource Class—Secure MXM Topics 721
Masking 722
Example: MXTOPIC resource class 722
NETCMDS Resource Class—Secure NETVIEW NETCMDS 723
Examples: NETCMDS resource class 724
NETSPAN Resource Class—Secure NETVIEW NETSPAN 725
Examples: NETSPAN resource class 726
NODES Resource Class— Control NJE Jobs and SYSOUT 727
Masking 728
Examples: NODES resource class 728
OPCMD Resource Class—Secure SVC 34 Commands 729
Example: OPCMD resource class 730
OPERCMDS Resource Class—Secure JES Operator Commands 731
Masking 732
Examples: OPERCMDS resource class 732
OTRAN Resource Class—Secure Ownable Transactions 733
Examples: OTRAN resource class 734
PANAPT Resource Class—Secure PANAPT for PANVELT 735
Example: PANAPT resource class 736
PANEL Resource Class—Secure Panels 737
Examples: PANEL resource class 738
PDSMEMn Resource Class—Secure Partitioned Data Set Members 739
Masking 740
Example: PDSMEMn resource class 740
PKISERV Resource Class—Control Access to R_Pkiserv Administrative Functions 740

Examples: PKISERV resource class 741
PPT Resource Class—Secure the CICS PPT 742
Example: PPT resource class 743
PROGRAM Resource Class—Secure Programs and Utilities 744
Protection 745
Examples: PROGRAM Resource Class 745
PROPCNTL Resource Class—Secure Special ACIDs 746
Examples: PROPCNTL resource class 746
PSB Resource Class—Secure the IMS Program Specification Block 747
Examples: PSB resource class 748
PSFMPL Resource Class—Secure Output Labeling Suppression 749
Examples: PSFMPL resource class 750
PTKTDATA Resource Class—PassTicket Data Resources 751
Examples: PTKDATA resource class 751
RECIPID Resource Class—Secure CADISPATCH RECIPID 752
Examples: RECIPID resource class 753
RESLIST Resource Class—Secure TSSAI RESLIST 754
Examples: RESLIST resource class 755
RODMMGR Resource Class—Secure NETVIEW RODMMGR 756
Examples: RODMMGR resource class 757
ROSRES Resource Class—Determine CA Roscoe CA—Vollie Command Ownership 758
Examples: ROSRES resource class 759
SCHEDULE Resource Class—Secure SCHEDULE for CA7 760
Examples: SCHEDULE resource class 761
SDSF Resource Class—Secure SDSF Commands 762
Masking 763
Example: SDSF Resource Class—Secure 763
SERVAUTH Resource Class—Secure TCP/IP Resources 764
Examples: SERVAUTH resource class 765
SERVER Resource Class—Secure z/OS SERVER Objects 766
Examples: SERVER resource class 767
SMESSAGE Resource Class—Secure TSO Messages 768
Examples: SMESSAGE resource class 769
SOMDOBJS Resource Class—Secure z/OS SOMDOBJS Objects 770
Examples: SOMDOBJS resource class 771
SPI Resource Class—Secure CEMT and CICS 772
Equivalents for CEMT INQUIRE and SET 773
Examples: SPI resource class 780
STATION Resource Class—Secure CADISPATCH and CASCHEDULER 781
Examples: STATION resource class 782
STORCLAS Resource Class—Secure SMS Storage Classes 783
Examples: STORCLAS resource class 784

SUBSCHEM Resource Class—Secure CAIDMS Subschema Names 785
Examples: SUBSCHEM Resource Class—Secure 786
SURROGAT Resource Class—Restrict Preset Security 787
Example: SURROGAT resource class 788
SYSAUTO Resource Class—IBM Automation Control for z/OS Resources 788
Examples: SYSAUTO resource class 789
SYSCONS Resource Class—Secure IBM Consoles 790
Examples: SYSCONS Resource Class 791
SYSMVIEW Resource Class—Secure SystemView Segments 792
Examples: SYSMVIEW resource class 793
TCICSTRN Resource Class—Secure CICS TCICSTRN RESCLASS 794
Examples: TCICSTRN resource class 795
TERMINAL Resource Class—Secure Terminals 796
Defining Terminals 796
Terminal Definitions for z/VM: 797
Terminal Definitions for z/OS 797
Terminal Definitions for PCs 798
Examples: TERMINAL resource class 799
TIMS Resource Class—Secure IMS TIMS 800
Examples: TIMS resource class 801
TOTAL Resource Class—Secure TOTAL for CINCOM 802
Examples: TOTAL resource class 803
TSOACCT Resource Class—Secure TSO Logon Account Codes 804
Examples: TSOACCT resource class 805
TSOAUTH Resource Class—Secure TSO User Attributes 806
Examples: TSOAUTH resource class 807
TSOPRFG Resource Class—Secure TSO Performance Groups 808
Examples: TSOPRFG resource class 809
TSOPROC Resource Class—Secure TSOP Logon Procs 810
Examples: TSOPROC resource class 811
TST Resource Class—Secure CICs Temporary Storage Table Names 812
Example: TST resource class 813
UR1/UR2 Resource Class—Secure Resource Restrictions 813
Customization 814
Examples: UR1/UR2 resource class 815
USRCLASS Resource Class—Secure Resource Classes 816
Examples: USRCLASS resource class 817
VMANAPPL Resource Class—Secure CAVMAN VMANAPPL 818
Example: VMANAPPL resource class 819
VMCF Resource Class—Secure VMCF Communication Targets 820
Examples: VMCF resource class 821
VMDEV Resource Class—Secure z/VM Real Devices 822

VMDIAL Resource Class—Secure Dial Access Virtual Machines 824
Examples: VMDIAL Resource Class—Secure 825
VMMACH Resource Class—Authorize AUTOLOG Users 826
Examples: VMMACH resource class 827
VMMDISK Resource Class—Secure Minidisks 828
ALL Access 829
Ownership 830
Masking 830
Examples: VMMDISK Resource Class—Secure 830
VMNODE Resource Class—Secure z/VM Nodes 831
Examples: VMNODE resource class 832
VMRDR Resource Class—Secure Virtual Machine Readers 833
Examples: VMRDR resource class 834
VOLUME Resource Class—Secure DASD and Tape 835
Vol/Ser Attributes 836
Examples: VOLUME resource class 836
VSELIB Resource Class—Secure TSSVSE Libraries 837
Example: VSELIB resource class 838
VSEPART Resource Class—Secure Partitions 839
Example: VSEPART resource class 840
VSEUSER Resource Class—Secure TSSVSE Users 841
Examples: VSEUSER resource class 842
VTAMAPPL Resource Class—Secure VTAM ACB Access 843
Examples: VTAMAPPL resource class 844
WRITER Resource Class—Secure Job Output 845
Masking 846
Examples: WRITER resource class 846
XCSFKEY Resource Class—Secure Exportation of ICSF Cryptographic Keys 847
Examples: XCSFKEY resource class 848
ZMFAPLA Resource Class—Secure z/OSMF Authorization Roles 849

Chapter 1: Introduction

This section contains the following topics:

Using Command Functions (see page 31)
Command Syntax (see page 31)
Entry Methods (see page 32)
Online Processing of Commands (see page 34)
Administrative Authority (see page 35)
Generic Prefixing (see page 35)

Using Command Functions

Security administrators use CA Top Secret command functions to communicate their administrative requirements to CA Top Secret. These requirements can range from the creation of an ACID to the definition of resource ownership.

CA Top Secret command functions are independent of the system facility. The security administrator uses command functions in the same manner, regardless of whether the facility is TSO, CICS, BATCH, CA-Roscoe®, IMS, or CA-IDMS®.

Command Syntax

CA Top Secret command syntax has the following format:

```
TSS FUNCTION(ACID|ACIDS|ALL|APPLU|AUDIT|DLF|FDT|MLSSTC|NDT|RDT|SDT)
    KEYWORD(OPERAND)[/* COMMENTS */]
```

TSS

CA Top Secret commands always begin with TSS.

FUNCTION

Specifies the function CA Top Secret performs. The rules for the function are:

- The function must immediately follow the TSS
- There can be one function only per TSS command
- One or more spaces must be entered between TSS and the function

ACID|ACIDS|ALL|APPLU|AUDIT|DLF|FDT|MLSSTC|NDT|RDT|SDT

Specifies the ACID being affected by the function.

KEYWORD

Specifies the resource type or security attribute being processed by the function. The rules for the keywords are:

- Keywords can be entered in any order.
- Keywords must be entered in full. Some keywords will not work if shortened.
- Keywords can be entered from line to line without special action.
- 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.

OPERAND

Specifies the prefix, resource name, required value, or name for a security attribute. The rules for operands are:

- Operands must be provided
- () is required to indicate no value
- If an operand is missing, any following keyword is ignored

COMMENTS

Enter comments at any point in a command after the first keyword.

The rules for comments are as follows:

- Comments must begin with /*.
- Comments should end with */, but this requirement may not apply to all cases.
- If a comment is not terminated with */ AND, the command line ends without a dash and the command and comment are considered complete.
- Comments may appear anywhere after the first KEYWORD and may be followed by additional keywords provided the closing */ exists.
- Comments are maintained on the command when logged to the Recovery File. This tool may be useful for documenting the reason the administration was performed.

Entry Methods

CA Top Secret functions can be entered freeform onto the command screen of an online terminal, or into any of the CA Top Secret administration panels.

Example: command syntax

This example creates the user USER01 with all of their required properties:

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

Access Administration Panels

TSS command functions can be entered and changed using the CA Top Secret full-screen administration panels, if the TSO installation uses IBM's System Productivity Facility (SPF or ISPF), or if the administrator is running under CMS. These panels provide the administrator with a *fill-in-the-blank* application for the TSS command.

To access the CA Top Secret selection panel

1. Access the ISPF/PDF Primary Option Menu.
2. Enter the option identifier corresponding to CA Top Secret security into the OPTION field of the ISPF Menu.

   The system displays the CA Top Secret Selection Panel:

```
P000001           Security Administration Main Menu          CA TOP SECRET
===>


Concerning Me...                        Resource Administration
  1  Who am I?                           21  Assign/remove resource ownership
  2  Lock my terminal                    22  Permit/revoke resource access
  3  Unlock my terminal                  23  Display resource access/ownership
                                         24  Certificate Management processing
ACID Administration
 11  Create ACID                        CA TOP SECRET System Administration
 12  Change ACID attributes              32  Modify security tables
 13  Assign administrative authority
 14  Display ACID information, REFRESH ACID
 15  Acid Compare/Modeling

Options for TSS Administration Session

   ( _ ) List after successful command
   ( _ ) Clear after successful command
   ( _ ) Display TSS command text

PF1= Help      2= Defaults 3= End       4= Return    5=           6=
PF7= Up        8= Down     9=          10=          11=          12=
```

Online Processing of Commands

When an ACID enters a CA Top Secret command function, CA Top Secret:

- Parses the entry for the correct syntax

- Determines if the ACID contains the proper administrative authority and scope to enter the command function
- Executes the command

Command Response Messages

Command functions cause CA Top Secret to issue a variety of messages. For information, see the *Messages and Codes Guide*.

If a command is successful, CA Top Secret issues the message:

TSS0300I *xxxx* FUNCTION SUCCESSFUL.

If a command fails CA Top Secret issues the message:

TSS0301I *xxxx* FUNCTION FAILED, RETURN CODE = *XX*

XX

Specifies the return code. Possible values are:

- 4—Syntax error or not authorized for this function
- 8—Functional Error (such as data set not found) or insufficient administration authority
- 16—Unexpected Error (message TSS0390E will also be issued)

Message TSS0301I is followed by a message in the TSS0200 series that explains the cause of the problem.

Administrative Authority

CA Top Secret processes only command functions (with the exceptions of HELP and WHOAMI) issued by ACIDs who have administrative authority. This administrative authority is limited to the scope of the administrator.

Generic Prefixing

Generic prefixing, designated with a (G), allows the administrator to identify multiple VMUSER IDs. This is used with:

- The PERMIT command
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ALL
- CPCMD(XAUTH)

Example: generic prefixing

This example permits a System's Programmer to use the spooling command CHANGE for files belonging to any user ID prefixed with TDG:

```
TSS PERMIT(SYSPROG) CPCMD(CHANGE)
                    VMUSER(TDG(G))
```

Chapter 2: Command Functions

ADDTO Function—Add Resource Ownership and Attributes

Valid on z/OS, z/VSE, and z/VM.

Use the ADDTO command function to perform the following activities:

- Assign resource ownership or special attributes to a defined ACID.
- Define started tasks to CA Top Secret.
- Add to the Audit Record a resource that CA Top Secret will audit.
- Authorize an ACID to use specific facilities.
- Connect profiles to users.
- Grant access to non-ownable installation-defined resources.
- Prohibit an ACID from using a specific set of commands or transactions (or confine an ACID to a specific set of commands or transactions).
- Add new resource classes to the Resource Descriptor Table (RDT).
- Add new fields to the Field Descriptor Table (FDT).
- Assign PassTickets, Node, NETUAF and volume records to the Node Descriptor Table (NDT).
- Identify which LUs can participate in APPC conversations by adding them to the APPCLU Record.

Issuing the ADDTO command function to suspend a signed-on ACID or add a profile to a signed-on ACID automatically triggers a type 71 RACF Event Notifications (ENF) signal, which provides notification about the change to the ACID's security record. Applications that receive the signal can take action (for example, immediately refresh the ACID's security record in a CICS remote region).

Administrators must have:

- Appropriate RESOURCE(OWN) authority to add ownership of resources from ACIDs within their scope
- MISC1, MISC2, or MISC9 authority to assign many of the security attributes to ACIDs within their scope

ACID authorities determine the levels at which administrators can manage ACIDs within their scope.

This command function has the following format:

```
TSS ADDTO(acid) keyword[(operand)]
```

acid

Specifies the ACID to which the resource or attribute is assigned.

keyword

Specifies a keyword that indicates the type of resource or attribute being assigned.

operand

Specifies the specific prefix, name, or value of resource or attribute being assigned. An operand might be required, depending on the specific keyword.

More information:

ADDTO Applicable Keywords (see page 40)

Resource Ownership

Resource ownership means that the user, profile, or control ACID has an access level of ALL. To avoid granting unlimited access to individual users or profiles:

- Use the ADDTO command function to assign resource ownership to department or division ACIDs
- Use the PERMIT command function to authorize full or restricted resource access for other ACIDs (non-owners)

If the resource is:

- New (undefined), the ADDTO command function defines the resource by assigning ownership to the ACID specified in the command function
- Owned by another ACID, the ADDTO command function transfers ownership to the ACID specified in the command function and permits the old owner full access to the resource

To permit access to all owned resources within a NOMASK resource class use the special resource name:

```
*ALL*
```

To permit resource classes with the MASK attribute use the resource name:

```
????
```

For classes with the MASK attribute:

- The minimum length of a resource that can be owned is two characters
- Ownership of a masking character cannot be defined unless the first character is a masking character

Examples: resource ownership

This example defines the resource by assigning ownership to the ACID:

```
TSS ADDTO(ACID1) DSNAME(USER)
```

This example transfers ownership of data set USER01 from ACID1 to ACID2:

```
TSS ADDTO(ACID2) DSNAME(USER01)
                 UNDERCUT
```

This example prevents automatic permission with the NOPERMIT keyword:

```
TSS ADDTO(ACID2) DSNAME(USER)
                 UNDERCUT
                 NOPERMIT
```

Assign Attributes

Adding an attribute to a user or profile ACID assigns special authorities or restrictions to that ACID.

Example: assigning an attribute

This example gives a user the ability to modify control options by adding the CONSOLE attribute:

```
TSS ADDTO(USER2) CONSOLE
```

ADDTO Applicable Keywords

This command function uses the keywords:

ACTION, ACTIVE, AFTER|BEFORE, ASSIZE, ASUSPEND, AUDIT, CALENDAR, CERTMAP, COMMAND, CONSOLE, CONVSEC, CNFAPP, CNFUVAR, CRITERIA, CRITMAP, DAYS, DAYS (For Calendars), DCDSN, DEFAULT, DEFNODES, DEFTKTLF, DFLTGRP, DIGICERT, DUFUPD, DUFXTR, EIMDOMAIN, EIMLOCREG, EIMOPTION, EIMPROF, EXPIRE, FACILITY, FIRST, FOR, GAP, GID, GROUP, HOME, ICSF, IDNFILTR, IESFL1, IESFL2, IESINIT, IESSYNM, IESTYPE, IESVCAT, IMSMSC, INSTDATA, INTERVAL, JOBNAME, KERBLINK, KERBNAME, KERBPASS, KERBUSER, KERBVIO, KEYRING, LABLCERT, LABLCMAP, LABLRING, LANGUAGE, LDAPDEST, LDS, LINKID, LINKNAME, LINUXNAM, LTIME, MASTFAC, MATCHLIM, MAXTKTLF, MCSALTG, MCSAUTH, MCSAUTO, MCSCMDS, MCSDOM, MCSKEY, MCSLEVL, MCSHC, MCSMFRM, MCSMGID, MCSMON, MCSROUT, MCSSTOR, MCSUD, MEMLIMIT, MINIKTLF, MMAPAREA, MODE, MRO, MULTIPW, NOADSP, NOATS, NODES, NODSNCHK, NOLCFCHK, NOPERMIT, NOPWCHG, NOREFRESH, NORESCHK, NOSUBCHK, NOSUSPEND, NOVMDCHK, NOVOLCHK, OECPUTM, OEFILEP, OIDCARD, OMVSPGM, OPCLASS, OPIDENT, OPPRTY, PASSWORD, PHYSKEY, PROCNAME, PROCUSER, PROFILE, PRXBINDDN, PRXBINDPW, PRXLDAPHST, PRXKRBREG, PRX509KRB, PSTKAPPL, PSUSPEND, REALMNAME, RESOWNER, RETAIN, RINGDATA, SCTYKEY, SDNFILTR, SESSKEY, SESSLOCK, SIGNMULTI, SHMEMMAX, SITRAN, SMSAPPL, SMSDATA, SMSMGMT, SMSSTOR, SNAME, SOURCE, START, STCACT, SUSPEND, SYSID, TARGET, THREADS, TIMEREC, TIME, TRACE, TRANSACTIONS, TRUST, TSOCOMMAND, TSODEFPRFG, TSODEST, TSOHCLASS, TSOJCLASS, TSOLACCT, TSOLPROC, TSOLSIZE, TSOMCLASS, TSOMPW, TSOMSIZE, TSOOPT, TSOSCLASS, TSOUDATA, TSOUNIT, TZONE, UID, UNAME, UNDERCUT, UNTIL, USAGE, USER, VSECATBT, VSEMCON, VSERDD, VSESYSAD, VSUSPEND, WAACCNT, WAADDR1, WAADDR2, WAADDR3, WAADDR4, WABLDG, WADEPT, WAIT, WAROOM, WANAME, XCOMMAND, XSUSPEND, and XTRANSACTIONS.

ADMIN Function—Grant Administrative Authority

Valid on z/OS, z/VSE, and z/VM.

Use the ADMIN command function to assign administrative capabilities to subordinate CA Top Secret administrators.

Administrative authority:

- Cannot be assigned to zones, division, department, or profile ACIDs.
- Can only be assigned to subordinate administrators within the administrator's scope. For example, VCAs cannot assign administrative authorities to other VCAs, even if they are in the same division. They can assign administrative authorities to DCAs, User ACIDs, and auditors within the divisions they control.

Only the MSCA defined to CA Top Secret during the installation process can create and assign administrative authorities to SCAs.

This command function has the following format:

```
TSS ADMIN(acid) keyword(authority level)
                ACCESS(access level)
```

acid

Specifies the ACID of the administrator being granted ADMIN authority.

keyword

Specifies the authority type the administrator is authorized to manage:

ACID

Specifies the authority levels, such as MAINTAIN, REPORT, AUDIT, and CREATE, at which administrators can manage ACIDs within their scope.

DATA

Information that the administrator may display using the TSS LIST function: BASIC, RESOURCE, XAUTH, LCF, SOURCE, PROFILE, INSTDATA, CICS, ADMIN, NAMES, PASSWORD, WORKATTR, SESSKEY, and ALL.

An administrator can display all of the above information, except SESSKEY, using the TSS LIST command function.

FACILITY

Any active facility contained in the Facility Matrix, such as VM, TSO, IMS, CICSTEST, and CA Roscoe.

MISC1

Low level administrative functions: LCF, INSTDATA, USER, LTIME, SUSPEND, NOATS, RDT, TSSSIM, and ALL.

MISC2

DLF, TARGET, NDT, TSO, SMS, APPCLU, WORKATTR, and ALL.

MISC3

SDT PTOK

MISC4

CERTSITE, CERTAUTH, CERTUSER, CERTLIST, CERTGEN, CERTEXPO, CERTCHEK, and KERBUSER

MISC5

DCLADMIN DCLLIST MLSADMIN SGVADMIN SGVLIST

MISC7

RSTDACC

MISC8

LISTRDT, LISTSDT, LISTSTC, LISTAPLU,MCS, PWMAINT,REMASUSP, and ALL.

MISC9

High level administrative functions: BYPASS, TRACE, CONSOLE, STC, MASTFAC, MODE, GLOBAL, GENERIC, and ALL.

RESOURCE

Allows administrators to give authority to issue commands for all resource types owned within their administrative scope. An administrator can also give authority for a specific resource (like DSNAME). All specific resources have the same authority levels; OWN, XAUTH, AUDIT, REPORT, INFO and ALL.

The access keyword, which is a subset of RESOURCE, is used with XAUTH to specify access levels. Access level operands depend on the type of resource to which access is being permitted. Examples include: CONTROL, CREATE, FETCH, NONE, READ, SCRATCH, UPDATE, WRITE, All .

resource

Used as an archetype for any resource defined to the RDT.

SCOPE

Used to define the scope for the LSCA

authority level

Specifies the authority level or levels at which the administrator will manage the authority type:

RESOURCE

OWN, XAUTH, AUDIT, INFO, REPORT, and ALL

ACID

XAUTH, AUDIT, CREATE, INFO, DEFNODES, REPORT, MAINTAIN, and ALL

DATA

BASIC, RESOURCE, XAUTH, LCF, SOURCE, INSTDATA, CICS, PROFILE, ADMIN, NAMES, ACID, WORKATTR, SESSKEY, PASSWORD,

and ALL

MISC1

LCF, INSTDATA, USER, LTIME, RDT, SUSPEND, NOATS, TSSSIM, and ALL

MISC2

SMS TSO, NDT, DLF, TARGET, WORKATTR, APPCLU, and ALL

MISC3

SDT PTOK

MISC4

CERTSITE, CERTAUTH, CERTUSER, CERTLIST, CERTGEN, CERTEXPO, CERTCHEK, and KERBUSER

MISC5

DCLADMIN DCLLIST MLSADMIN SGVADMIN SGVLIST

MISC7

RSTDACC

MISC8

LISTRDT, LISTSTC, LISTAPLU, MCS, LISTSDT, PWMAINT, REMASUSP, and ALL

MISC9

BYPASS, TRACE, CONSOLE, STC, MASTFAC, MODE,GLOBAL, GENERIC, and ALL

FACILITY

Specific Facilities

SCOPE

Define scope for LSCA

access level

Specifies the access levels (for example, FETCH and WRITE) for which the administrator is authorized to PERMIT RESOURCE(XAUTH) resource access. If no entry is made, CA Top Secret usually assigns a default level based on the resource type.

Access Levels for RESOURCE(XAUTH) only

CONTROL, CREATE, FETCH, NONE, READ, SCRATCH, UPDATE, WRITE, All

Examples: ADMIN function

In this example the security administrator VCA1 assigns ownership of data sets by adding them to the Audit Record:

```
TSS ADDTO(VCA1) DSNAME(OWN,AUDIT)

TSS ADDTO(DEPT1) DSNAME(XYZ.DATA.CRASH)
```

This example audits the specified data set:

```
TSS ADDTO(AUDIT) DSNAME(XYZ)
```

This example gives all users the ability to LIST their BASIC data, resource permissions, LCF transactions and password data, without allowing them to manipulate their data:

```
TSS ADMIN(ALL) DATA(BASIC,XAUTH,LCF,PASSWORD)
```

CHKCERT Function—Check Certificates

Valid on z/OS.

Use the CHKCERT command function to display information about digital certificates. For example, you can display information about whether a digital certificate in a data set (DCDSN) is in the CA Top Secret security file and associated with an ACID.

The administrator must have MISC4(CERTCHEK) *or* UPDATE access to TSSCMD.CERTUSER.CHKCERT in the CASECAUT resource class.

This command function has the following format:

```
TSS CHKCERT DCDSN(input_dataset_name)
            PKCSPASS('pkcs#12_password')
            CHAIN
```

DCDSN(*input_dataset_name*)

Specifies the data set in which the digital certificate exists.

pkcs#12_password

(Required if the data set contains a PKCS#12-formatted certificate that is password-protected) Specifies a case-sensitive PKCS password that can also contain blanks. The passwords associated with PKCS #12 certificates are not viewable. It is the CA Top Secret administrator's responsibility to keep track of the PKCS #12 password assigned to the digital certificate.

Range: Up to 255 characters

CHAIN

(Optional) Displays information for each certificate in the chain of the input data set and displays the following summary information as applicable:

- Number of certificates in the chain
- Whether the chain is complete or incomplete
- Whether the chain contains expired or non-trusted certificates
- Whether any certificate in the data set is not present in the CA Top Secret database

Example: Display Certificate Associations

This example uses the DCDSN keyword to specify a certificate package:

```
TSS CHKCERT DCDSN(reipa02.user2.cert2)
```

The product reviews each certificate in the package. If a certificate is in the database, the product lists the user and ID to which the certificate is defined.

COMPARE Function—Compare ACIDs

Valid on z/OS.

Use the COMPARE command function (and the USING keyword) to compare the security records of two ACIDs.

Note: Administrators must have explicit authority to list a TSS data type of ACIDS. This authority is obtained through the DATA keyword of the ADMIN command function. The comparison does not include special security records (for example, the ALL Record, the FDT record, or the STC record); does not include digital certificate information; and does not let you compare creation dates, administrative changes dates, and last used dates. For more information about special security records, see the *CA Top Secret User Guide.*

This command function has the following format:

```
TSS COMPARE(acid1) USING(acid2)
```

acid1

Specifies the first ACID to compare.

acid2

Specifies the second ACID to compare.

Example: Compare the Security Records of ACIDs Rachel01 and Rachel02

This example compares the security records of ACIDs rachel01 and rachel02:

```
TSS COMPARE(rachel01) USING(rachel02)
```

More information:

USING Keyword with COMPARE—Specify ACID to Compare (see page 531)

CREATE Function—Define a New ACID

Valid on z/OS, z/VSE, and z/VM.

Use the CREATE command function to define new ACIDs to CA Top Secret. The CA Top Secret administrator can also assign resource ownership and/or security attributes while creating the ACID.

The administrator must have the following authority:

- ACID(CREATE) authority, through the TSS ADMIN function, to create ACIDs within the administrator's scope
- RESOURCE(OWN) authority, through the TSS ADMIN function, to assign resource ownership to ACIDs within the administrator's scope
- MISC1, MISC2, and MISC9 authorities, through the TSS ADMIN function, to assign security attributes

This command function has the following format:

```
TSS CREATE(acid)
```

acid

Specifies the name of the ACID that you want to create. The following characters are allowed in the ACID name:

- Alphabetical: A-Z
- Numeric: 0-9
- National: $ # @ % & = ?
- - + ¢ | < ! ; ¦ > : { } ¬ (allowed only for ACIDs that cannot sign on (type ZONE, DIVISION, DEPARTMENT, PROFILE, GROUP))

Example: Define an ACID

This example creates the definition for ACID CLRK99:

```
TSS CREATE(CLRK99) TYPE(USER)
                   NAME('Bill Smith')
                   PASS(Bosco)
                   DEPT(LEVEL01)
```

Use of DEPARTMENT, DIVISION, and ZONE

The following describes how SCAs, LSCAs, ZCAs, VCAs, and DCAs enter the DEPARTMENT, DIVISION, and ZONE keywords:

SCA

SCAs must enter the ZONE, DIVISION or DEPARTMENT keyword with all TSS CREATE entries which require these keywords.

LSCA

LSCAs must enter the ZONE or DIVISION keyword with all TSS CREATE entries which require these keywords.

ZCA

ZCAs cannot enter the ZONE keyword in their CREATE entries. CA Top Secret automatically assigns the ACID to the ZCA's zone. The ZONE keyword is required if the person entering the command function is an SCA.

VCA

VCAs cannot enter the DIVISION keyword in their CREATE entries. CA Top Secret automatically assigns the ACID to the VCA's division. The DIVISION keyword is required if the person entering the command function is an SCA.

DCA

DCAs cannot enter the DEPARTMENT keyword in their CREATE entries. CA Top Secret automatically assigns the ACID to their department. The DEPARTMENT keyword is required if the person entering the command function is not a DCA.

CREATE Applicable Keywords

The NAME must be entered as part of all CREATE functions.

Depending on the type of ACID being created, the TYPE keyword as well as the DEPARTMENT and/or DIVISION keywords are also specified.

The command function uses the keywords:

ABSTRACT, APPLICATION, AREA, ASUSPEND, AUDIT, COMMAND, CONSOLE, CPCMD, CACMD, CPU, DATABASE, DBD, DB2, DB2BUFFP, DB2COL, DB2DBASE, DB2PKG, DB2PLAN, DB2STOGP, DB2SYS, DB2TABLE, DB2TABSP, DCSS, DCT, DEVICES, DIAGNOSE, DIVISION, DSNAME, DUFUPD, DUFXTR, FACILITY, FCT, FIELD, FOR, GAP, GROUP, IBMFAC, IBMGROUP, INSTDATA, IUCV,, JCT, JESJOBS, JESPOOL, KERBVIO, LANGUAGE, LTIME, MASTFAC, MATCHLIM, MGMTCLAS, MODE, MRO, MULTIPW, NAME, NOADSP, NODES, NODSNCHK, NOATS, NOLCFCHK, NOPERMIT, NOPWCHG, NORESCHK, NOSUBCHK, NOSUSPEND, NOVMDCHK, NOVOLCHK, OIDCARD, OPCLASS, OPCMD, OPERCMDS, OPIDENT, OPPRTY, OTRAN, PANEL, PASSWORD, PHYSKEY, PPT, PROCNAME, PROFILE, PROGRAM, PROPCNTL, PSB,, PSFMPL, RSTDACC, SCTYKEY, SDSF, SITRAN, SMESSAGE, SMSAPPL, SMSDATA, SMSMGMT, SMSSTOR, SOURCE, SPI, STCACT, STORCLAS, SUBSCHEM, SUSPEND, TARGET, TERMINAL, TRACE, TRANSACTIONS, TSOACCT, TSOAUTH, TSOCOMMANDS, TSODEFPRFG, TSODEST, TSOHCLASS, TSOJCLASS, TSOLACCT, TSOLPROC, TSOLSIZE, TSOMCLASS, TSOMPW, TSOMSIZE, TSOOPT, TSOPRFG, TSOPROC, TSOCLASS, TSOUDATA, TSOUNIT, TSTTYPE, TZONE, UNDERCUT, UNTIL, UR1/UR2, USER, USING, USRCLASS, USING, VMCF, VMDIAL, VMMACH, VMMDISK, VMNODE, VMRDR, VOLUME, VTAMAPPL, VXDEVICE, VXFILE, WAACCNT, WAADDR1, WAADDR2, WAADDR3, WAADDR4, WABLDG, WADEPT, WAIT, WANAME, WAROOM, DUFXTR OPPRTY TSOLSIZE WRITER, FACILITY OTRAN TSOMCLASS XCOMMAND, and ZONE.

DEADMIN Function—Remove Administrative Authority

Valid on z/OS, z/VSE, and z/VM.

Use the DEADMIN command function to revoke administrative capabilities from subordinate CA Top Secret administrators.

Administrative authority:

- Cannot be assigned to zones, division, department, or profile ACIDs.
- Can only be used revoke administrative authorities from subordinate administrators within the administrator's scope. For example, VCAs cannot revoke administrative authorities from other VCAs, even if they are in the same division. VCAs can revoke administrative authorities from DCAs, user ACIDs, and auditors within the divisions they control.
- Only the MSCA defined to CA Top Secret during installation can revoke administrative authorities to SCAs.
- Can only be deadministered for authorities the command issuer already controls.

For information on the format, rules, and restrictions that apply to the DEADMIN function, see the ADMIN command function.

Examples: DEADMIN command function

This example removes VCA1's authority to assign the AUDIT attribute for DSNAMEs:

```
TSS DEADMIN(VCA1) DSNAME(AUDIT)
```

VCA1 still has the authority to assign ownership of DSNAMEs to ACIDs within his scope.

This example totally removes VCA1's authority for DSNAMEs:

```
TSS DEADMIN(VCA1) DSNAME(ALL)
```

DELETE Function—Delete an ACID

Valid on z/OS, z/VSE, and z/VM.

Use the DELETE command function to remove an ACID definition from the security file.

You *cannot* delete the following ACIDs:

- An ACID to which other ACIDs connected. Division, department, and/or profile ACIDs cannot be deleted if other ACIDs are connected to them. To delete a department or division ACID, use the TSS MOVE command function to transfer all ACIDs out of the department or division being deleted. To delete a profile ACID, use the TSS REMOVE command function to remove the profile from any connected user ACIDs.
- An ACID that owns resources permitted to other ACIDs.
- Special ACIDs such as ALL, STC, AUDIT, DLF, RDT, SDT, or the MSCA's ACID.

If an ACID is deleted, CA Top Secret:

- Disconnects the deleted ACID from all profiles it is connected to
- Revokes any access permissions that the ACID was granted
- Removes any resources that the ACID owns

Issuing the DELETE command to delete a signed-on ACID automatically triggers a type 71 RACF Event Notifications (ENF) signal, which provides notification about the change to the ACID's security record. Applications that receive the signal can take action (for example, immediately remove the ACID's security record in a CICS remote region).

Administrators must have ACID(CREATE|ALL) authority, through the ADMIN function, to delete ACIDs within their scope.

This command function has the following format:

```
TSS DELETE(acid)
```

acid

Specifies the ACID being deleted.

Example: Delete an ACID

This example removes the definition for ACID CLRK99:

```
TSS DELETE(CLRK99)
```

More information:

MOVE Function—Move an ACID (see page 69)
REMOVE Function—Remove Resource Ownership or Resources (see page 89)

EXPORT Function—Write a Certificate to a Data Set

Use the EXPORT command function to export digital certificates to a new data set (DCDSN) after it has been added to a user. From there it can be exported to a browser or transferred to another platform.

The DIGICERT name or the LABLCERT label name must be entered as part of all EXPORT functions since these keywords indicate the certificate being exported.

EXPORT by default, gives you the certificate and the public key. When you do an EXPORT with either one of the PKCS#12 package formats (PKCS12DER or PKCS12B64) you get the certificate, public key and any inherited signing certificates.

The EXPORT DCDSN (data set) must not be defined or the output DCDSN cannot be allocated or cataloged.

Administrators must have:

- ACID(MAINTAIN) for users within their scope
- MISC4(CERTEXPO) and MISC4(CERTSITE) for CERTSITE ACID
- MISC4(CERTAUTH) for CERTAUTH ACID
- Administrators without the above authorities can issue the EXPORT command if they have:
  - UPDATE access to TSSCMD.CERTUSER.EXPORT in the CASECAUT resource class when the certificate is associated with a user ACID
  - UPDATE access to TSSCMD.CERTSITE.EXPORT in the CASECAUT resource class when referencing a site certificate
  - UPDATE access to TSSCMD.CERTAUTH.EXPORT in the CASECAUT resource class when referencing a certificate-authority certificate

This command function has the following format:

```
TSS EXPORT(acid) [DIGICERT(name)|LABLCERT(label-name)]
                 DCDSN(output-data-set-name)
                 FORMAT(format type)
                 PKCSPASS(PKCS#12 password)
```

acid

Specifies the ACID for the digital certificate being exported.

DIGICERT(*name*)

Specifies the keyword that identifies the digital certificate being exported.

LABLCERT(label-name)

Specifies the keyword that identifies digital certificate label name being exported.

DCDSN(*output-data-set-name*)

Specifies the data set name the digital certificate is written to.

FORMAT(format type)

Specifies the format; CERTB64, CERTDER, PKCS12B64, PKCS12DER, PKCS7B64, or PKCS7DER.

PKCSPASS(PKCS#12 password)

Defines the PKCS#12 formats password when FORMAT(PKCS12*nnn*) is used.

GENCERT Function—Generate a Certificate

Valid on z/OS, z/VSE, and z/VM.

Use the GENCERT command function to generate a digital certificate and insert a CERTDATA profile record into the CA Top Secret info-storage database.

Specify a DIGICERT name as part of all GENCERT functions since the DIGICERT keyword indicates the name used in the digital certificate.

Administrators must have:

- ACID(MAINTAIN) and MISC4(CERTGEN) for users within their scope
- MISC4(CERTSITE) for CERTSITE ACID
- MISC4(CERTAUTH) for CERTAUTH ACID
- Administrators without the above authorities can issue the GENCERT command if they have:
  - UPDATE access to TSSCMD.CERTUSER.GENCERT in the CASECAUT resource class when the certificate is associated with a user ACID
  - UPDATE access to TSSCMD.CERTSITE.GENCERT in the CASECAUT resource class when generating a site certificate
  - UPDATE access to TSSCMD.CERTAUTH.GENCERT in the CASECAUT resource class when generating a certificate-authority certificate

This command function has the following format:

```
TSS GENCERT [{CERTAUTH|CERTSITE|acid}]
            DIGICERT(8-byte-name)
            {DCDSN(request-data-set-name)\}
            {SUBJECTN ('CN="common-name"
                       T="title"
                       OU="org-unit-name1,org-unit-name2"
                       O="organizational-name"
                       L="locality"
                       ST="state-or-province"
                       C="2–digit–only country code"')}
            [NBDATE(mm/dd/yy) NBTIME(hh:mm:ss)]
            [NADATE(mm/dd/yy) NATIME(hh:mm:ss)]
            [KEYSIZE(160-4096))
            [LABLCERT(label-name)]
            [ICSF|PCICC|DSA|NISTECC|BPECC]
            [SIGNWITH(acid,digicert)]
            [SIGNALG(SHA1|SHA256)]
            [KEYUSAGE(HANDSHAKE DATAENCRYPT DOCSIGN CERTSIGN KEYAGREE)]
            [ALTNAME('IP=numeric-IP-address
```

```
                    DOMAIN=internet-domain-name
                    EMAIL=email-address
                    URI=universal-resource-identifier')]
```

Note: Include single quotes if specifying more than one value with KEYUSAGE. For example:

```
KEYUSAGE('HANDSHAKE DATAENCRYPT')
```

If DCDSN or SUBJECTN is not specified, the SUBJECTN defaults to the ACID's name field.

The three types of certificates that you can specify with GENCERT are:

CERTAUTH

Designates the certificate as a certificate-authority certificate.

CERTSITE

Designates the certificate as a site certificate.

acid

Designates the user associated with the certificate.

GENCERT Applicable Keywords

The command function uses the keywords:

- ALTNAME
- DIGICERT
- DCDSN
- ICSF/PCICC/DSA/NISTECC/BPECC
- KEYSIZE
- KEYUSAGE
- LABLCERT
- NADATE
- NATIME
- NBDATE
- NBTIME
- SIGNWITH
- SIGNALG
- SUBJECTN

GENREQ Function—Generate a Certificate Request

Valid on z/OS, z/VSE, and z/VM.

Use the GENREQ command function to generate a PKCS#10 base64-encoded digital certificate request and write it to a data set. This request contains the subject's distinguished name and public key, and is signed with the private key associated with the specified certificate.

This command generates comments at the beginning of the certificate. Delete the comments if the application accepting the certificate does not support comments.

These requests can be sent to a third-party certificate authority, such as Verisign, or they can be imported into and signed by CA Top Secret using the TSS GENCERT command function. The private key is then assigned by the third-party authority, or when input into the TSS GENCERT command.

Specify the DCDSN keyword and the DIGICERT name or the LABLCERT label name since these keywords indicate the name used in the digital certificate. To use keyword GENREQ, the ACID and DIGICERT|LABLCERT must already exist. DIGICERT or LABLCERT is used to locate the certificate.

Administrators must have:

- ACID(MAINTAIN) and MISC4(CERTGEN for users within their scopeKEY
- MISC4(CERTSITE) for CERTSITE ACID
- MISC4(CERTAUTH) for CERTAUTH ACID
- Administrators without the above authorities can issue the GENREQ command if they have:
  - UPDATE access to TSSCMD.CERTUSER.GENREQ in the CASECAUT resource class when the certificate is associated with a user ACID
  - UPDATE access to TSSCMD.CERTSITE.GENREQ in the CASECAUT resource class when referencing a site certificate
  - UPDATE access to TSSCMD.CERTAUTH.GENREQ in the CASECAUT resource class when referencing a certificate-authority certificate

This command function has the following format:

```
TSS GENREQ(acid) DCDSN(output-data-set-name)
                 DIGICERT(name) | LABLCERT(label-name)
```

acid

The record key of the certificate to use to obtain the distinguished name and public key for the request, if LABLCERT is not also specified. This may be a one-to-eight character acid. If LABLCERT is specified, userid must be specified, and indicates the acid that the label is associated with.

HELP Function—Request Information

Use the HELP command function to provide basic information about the use of each TSS command function.

This command function has the following format:

```
TSS HELP OPERAND(function)
```

Note: The HELP command cannot be routed through the security network using the Command Propagation Facility (CPF).

The function must equal one of the following CA Top Secret command functions:

ADDTO	LIST	REMOVE	WHOAMI
ADMIN	LOCK	RENAME	WHOHAS
CREATE	MODIFY	REPLACE	WHOOWNS
DEADMIN	MOVE	REVOKE	
DELETE	PERMIT	UNLOCK	

When the HELP command function is entered onto the screen, CA Top Secret responds with a list of all keywords that may be used with the command function entered in the command.

Example: HELP function

This example obtains information concerning the ADMIN function:

```
TSS HELP OPERAND(ADMIN)
```

LIST Function—Display ACID Security Data

Valid on z/OS, z/VSE, and z/VM.

Use the LIST command function to display data from the following records:

- Security record of a specific ACID
- Security record of all ACIDs that match a specific prefix
- Security record of all ACIDs of a specific type
- Security record of all ACIDs in a department and/or division
- AUDIT, STC, NDT, RDT, FDT, DLF, SDT, APPCLU, and/or ALL records

After the ACIDs have been specified in the command, use the DATA operand to limit the output. Use the DATA(TERSE) operand to limit the security file I/O.

TSS LIST entries vary, depending on the type of data being requested and the security record from which data is obtained. Administrators must have explicit authority (through the ADMIN - DATA command function) to list TSS data types.

CA Top Secret displays only data concerning ACIDs within the administrator's scope. An MSCA or an authorized SCA can list data for the entire site. A VCA may list data for his or her division (and all subordinate departments), and a DCA may list data for his or her department. When a DCA, VCA, or ZCA lists a user within their scope, the DEPARTMENT, DIVISION, and ZONE are included in the display.

ACIDs are listed first by division, then by the department within the division. User and profile ACIDs are listed in alphabetical order within an organizational grouping.

This command function has the following format:

TSS LIST (*acid*) *keyword*(*setting*)

acid

Specifies the ACID for which you are displaying security data. Specifying ACIDS lists data for all ACIDs that match the accompanying keyword criteria. For example, you can specify TSS LIST(ACIDS) with the DEPARTMENT keyword (and other keywords) to obtain data about all ACIDs in a department.

***keyword*(*setting*)**

Specifies an available keyword (and a setting for that keyword if applicable). For example, you can you can include ACIDPRFX(TEST) as part of the LIST command, where the TEST setting specifies to list all ACIDs that begin with *TEST*.

This command function uses the keywords:

- ACIDPRFX
- ARCHIVE
- CHAIN
- DATA
- DEPARTMENT
- DIGICERT

  Note: TSS LIST DIGICERT does not display the ALTNAME parameter. Only the CHKCERT command displays this parameter.
- DISPLAY

- DIVISION
- FDTNAME
- FDTCODE
- INTO
- ISSUERDN
- KEYRING
- LABLCERT
- LABLRING
- LINKID
- PREFIX
- PSTKAPPL
- RESCLASS
- RESCODE
- SEGMENT
- SERIALNUM
- SESSKEY
- TARGET
- TYPE
- ZONE

Example: List All Data for a Specific ACID

This example obtains all the data about a specific ACID and contents of all profiles connected to that ACID:

```
TSS LIST(acid) DATA(PROFILE,ALL)
```

Example: List Data for ACIDs with a Specific Prefix

This example obtains all the data about the ACIDs starting with a specific prefix and the contents of all profiles connected to those ACIDs:

```
TSS LIST(ACIDS) ACIDPRFX(acid_prefix)
                DATA(PROFILE,ALL)
```

Note: Prefixes can be from one to seven characters long.

Example: List Data for All ACIDs of a Specific Type

This example obtains data about all ACIDs of a specific type:

```
TSS LIST(ACIDS) TYPE(USER | PROFILE | GROUP | DCA | VCA | DEPARTMENT |
                     DIVISION | SCA | LSCA | ZONE | ZCA)
                DATA(BASIC,RESOURCE,XAUTH,LCF,SOURCE,INSTDATA,CICS,ADMIN,NAMES,
                     TSO,ACIDS,EXPIRE,PASSWORD|ALL[,PROFILE,PASSWORD,EXPIRE])
```

Example: List Data for All ACIDs in a Department

This example obtains data about all ACIDs in a department:

```
TSS LIST(ACIDS) DEPARTMENT(department acid)
                DATA(BASIC,RESOURCE,XAUTH,LCF,SOURCE,TSO,
                INSTDATA,CICS,ADMIN,NAMES,ACIDS,
                PASSWORD|ALL [,EXPIRE,PASSWORD,PROFILE])
```

Example: List Data for All ACIDs in a Division

This example obtains data about all ACIDs in a division:

```
TSS LIST(ACIDS) DIVISION(div. acid)
                DATA(BASIC,RESOURCE,XAUTH,LCF,SOURCE,TSO,
                INSTDATA,CICS,ADMIN,NAMES,ACIDS,
                PASSWORD|ALL [,EXPIRE,PASSWORD,PROFILE])
```

Example: List Data for All ACIDs in a Zone

This example obtains data about all ACIDs in a zone:

```
TSS LIST(ACIDS) ZONE(zon. acid)
                DATA(BASIC,RESOURCE,XAUTH,LCF,SOURCE,TSO,
                INSTDATA,CICS,ADMIN,NAMES,ACIDS,
                PASSWORD|ALL [,EXPIRE,PASSWORD,PROFILE])
```

Example: List Contents of the ALL Record

This example obtains data about the contents of the ALL record:

```
TSS LIST(ALL)
```

Example: List Contents of the DLF Record

This example obtains data about the contents of the DLF record:

```
TSS LIST(DLF)
```

Example: List Contents of the AUDIT Record

This example obtains data about the contents of the AUDIT record:

```
TSS LIST(AUDIT)
```

Example: List Contents of the STC Record

This example obtain data about the contents of the STC record:

```
TSS LIST(STC)
```

Example: List Contents of the RDT Record

This example obtains data about the contents of the RDT record:

```
TSS LIST(RDT)
```

Example: List Contents of the FDT Record

This example obtains data about the contents of the FDT record:

```
TSS LIST(FDT)
```

Example: List Contents of the NDT Record

This example obtain data about the contents of the NDT record:

```
TSS LIST(NDT)
```

Example: List Contents of the SDT Record

This example obtains data about the contents of the SDT record:

```
TSS LIST(SDT)
```

Example: List Contents of the APPCLU Record

This example obtains data about the contents of the APPCLU record:

```
TSS LIST(APPCLU)
```

Example: List Contents of the MLS Record

This example obtains data about the contents of the MLS record:

```
TSS LIST(MLS)
```

Example: List Profile ACIDs Associated with USER01 by Order of Processing

This example produces a list of the profile ACIDs associated with USER01 in the order in which they are searched by the Security Algorithm:

```
TSS LIST(USER01) DATA(PROFILES,NOSORT)
```

Hard Copy Listings

Hard copy listings are obtained by using Batch TMP of TSSCFILE in z/OS and the batch utility TSSCRIPT for z/VM.

ACID Types

Use the following list to translate the ACID types that appear on the output of a TSS LIST command:

Code	ACID type
DC	DCA
D	Department
V	Division
LC	LSCA
P	Profile
SC	SCA
ZC	ZCA
VC	VCA
Z	Zone

Profile Sorting

When a TSS LIST(acid) DATA(PROFILES) command is issued, the profiles associated with that ACID is listed alphabetically.

To view the profiles in the order in which they are processed, enter:

```
TSS LIST(acid) DATA(PROFILES,NOSORT)
```

ACID Lists

When a TSS LIST(acid) DATA(BASIC) command is issued, the output associated with that ACID is sorted alphabetically (for example, LOCK TIME comes before TIME ZONE).

To list the segments and the fields within the segments alphabetically, enter:

```
TSS LIST(acid) SEGMENT(ALL)
```

Groupings are identified by the segment which they are in, not a header line. For example, groupings that were separated by a header line such as TSO DATA are now identified as SEGMENT TSO.

LIST in a Shared Environment

In a shared security file environment, all modifications to any SDT record type and data field made in the local system are immediately available to the TSS LIST command if entered from the same local system.

If the SDT modifications are made on a local system and the TSS LIST is attempted from a remote system, it is possible that some SDT records will not reflect the current changes as they are listed from internal tables not updated with the current data. The SDT record types and data fields that may experience this effect include:

- CERTMAP Certificate name filter
- CRITERIA Certificate filter criteria
- CRITMAP Certificate filter ACID to user
- DIGICERT Digital certificates
- KERBLINK Kerberos foreign principals
- KERBSEGM Kerberos principal user
- KEYRING Certificate keyring
- LINUXNAM Linux user name
- REALM Kerberos local / foreign realms

For immediate access to the current TSS LIST record data for any of the included SDT record types on a remote system, enter the TSS MODIFY SYNCH command to refresh the required tables with current data from the security file.

LOCK Function—Lock Online Terminal

Valid on z/OS, z/VSE, and z/VM.

Use the LOCK command function to lock a terminal so it cannot be used when unattended. The LOCK function is disabled if the NOLCFCHK attribute is attached to the ACID.

An attempt to use a locked terminal in:

- z/VM, IMS, CA-IDMS/UCF, and CA-Roscoe results in immediate termination of the command or transaction.
- CICS and CA-IDMS/DC results in a prompt for the ACID's password. If the password is valid, the request is executed; if not, the request is terminated.

In TSO, the command takes affect when going from one program to another. In ISPF/EDIT, you can still use ACCESS and ALLOCATE with data sets to which you have access. The TSS LOCK command takes affect when you leave EDIT/BROWSE.

The LOCK command cannot be routed through the security network using the Command Propagation Facility (CPF).

This command function has the format:

```
TSS  LOCK
```

MODEL Function—Create a New User from a Model

Use the MODEL command function to generate the TSS commands to create an acid of the same type (using an existing acid as a model). The MODEL command generates a list of TSS commands, which you can review and edit as necessary. Executing the commands creates the acid specified in the ACID keyword.

Note: If you use the MODEL command, and the DIAGSYM option in DFSORT is YES, you must have a SYSOUT DD in the CA Top Secret started task.

The generated commands represent most of the information available from a TSS LIST(modelacid) DATA(ALL) command. All attributes and data set and resource permissions are modeled by commands. Fields that are not modeled include:

- Passwords or passphrases
- Digital certificate and keyring segments:
- Create/Modify date time

- Last used info
- ACID List entries
- ADMINBY
- ASUSPEND/VSUSPEND/PSUSPEND
- Data Set Ownership information
- Resource Ownership information

The INTO keyword specifies the data set where the TSS commands are written. You must specify this keyword to write commands to a PDS data set. You can view or edit the generated TSS commands.

If the INTO keyword is not specified, the TSS commands that are produced by the MODEL command are displayed to the user who issued the command.

The administrator executing the command must have:

- Admin authority for DATA(all)
- Scope over the acid being modeled

When the MODEL command is executed, the first record in the output is a comment containing the following information:

- The command
- The user acid being modeled
- The date and time of the model
- Identification of the TSS administrator who issued the command
- Identification of the system on which the command was executed
- Identification of the user acid used as a model

An example comment is as follows:

```
/*MODEL ACID NEWCAS   STORED 02/18/10-10.45.51 BY MASTER1  ON XE15 USING CASSIE
```

This keyword has the following format:

```
TSS MODEL USING(modelacid) ACID(newacid)
TSS MODEL USING(modelacid) ACID(newacid) [INTO(datasetname[(membername)])]
```

modelacid

Specifies the acid to use as a model. All resources and permissions from this acid are copied to the new acid.

newacid

Specifies the name of the acid that will be modeled after the acid specified in the USING keyword. The name can be a new acid name or an existing acid name. If you specify an existing acid name and want to use TSS commands produced by the MODEL commands without changing them, delete the existing ACID before executing the TSS commands.

***datasetname*[(*membername*)]**

(Optional) Specifies the name and optionally a member name of the PDS data set where the model output commands are stored. If the keyword INTO is not specified, the TSS commands produced by the MODEL command are displayed to the user who issued the MODEL command.

Default for *membername***:** Value specified in the ACID keyword. However, if the first character of the ACID value is a number, you must supply a member name that does not start with a number.

Examples: MODEL function

- This example generates TSS model command output for the new acid NewRac using existing acid Rachael. The specified command generates the output in a user-specified member in a MVS PDS data set named KOTPA01.RACHAEL.MODEL (member name Rachaelk).

  ```
  TSS MODEL USING(Rachael) ACID(NewRac) INTO(kotpa01.rachael.model(Rachaelk))
  ```

- This example generates the model command output for the new acid NewRac using existing acid Rachel in the MVS PDS data set named KOTPA01.RACHAEL.MODEL (without a member name). The member is set to the acid name in the ACID keyword NewRac.

```
TSS MODEL USING(Rachael) ACID(NewRac) INTO(kotpa01.rachael.model)
```

- In this example, the INTO keyword is omitted, and the command output appears on screen of the user (or appears as batch output if the command is executed from batch):

  ```
  TSS MODEL USING(cassie) ACID(newcas)
  ```

  Command output:

  ```
  /*MODEL ACID NEWCAS   STORED 02/18/10-14.41.53 BY MASTER1  ON XE15 USING CASSIE
  /*Please edit any CREATE commands by adding a PASSWORD keyword to the command
  TSS CRE(NEWCAS) NAME('CASSIE E. KOT') TYPE(USER) DEPT(SYSDEPT)
  TSS ADD(NEWCAS) GROUP(OMVSGRP)
  TSS ADMIN(NEWCAS) MISC4(CERTAUTH CERTUSER CERTGEN  CERTEXPO CERTCHEK)
  TSS ADD(NEWCAS) FAC(CICSPROD)
  TSS ADD(NEWCAS) FAC(TSO)
  TSS ADD(NEWCAS) KERBNAME(CASSIE1PRINCIPAL)
  TSS ADD(NEWCAS) MAXTKTLF(0000086400)
  TSS ADD(NEWCAS) HOME(/U)
  TSS ADD(NEWCAS) DFLTGRP(OMVSGRP)
  TSS0300I  MODEL    FUNCTION SUCCESSFUL
  ```

Fields That Are Not Copied During User Modeling

During the user modeling process, CA Top Secret copies most of the modeled user's security record information; however, the following fields are *not* copied during the process:

- Field values not displayed by a TSS LIST command
- Passwords or passphrases
- Digital certificate and keyring segments:
  - Certificate name, certificate start date, certificate until date, certificate ID, certificate subject DN
  - Certificate keyring, certificate serial number, certificate issuer IDN, certificate issuer SDN, certificate NB date
  - Certificate NA date, certificate key size, certificate key type, certificate label, certificate trust status
  - Certificate URI, certificate IP address, certificate key usage
- Create/Modify date
- Create/Modify time
- Last used info
- ACID List entries

- ADMINBY
- Facility ADMINBY
- Segment start
- FCT ownership
- Ownership dsn
- Ownership vol
- Ownership vm
- Ownership resource
- Ownership pie resource
- Ownership resource class
- Ownership on WHOHAS

MODIFY Function—Display or Alter Control Options

Use the MODIFY function to perform the following activities:

- Display the status of the global security environment.
- Enter, change, or display CA Top Secret control options.

MODIFY requests must be made from an online terminal (for example, TSO, CICS, IMS/DC) in a z/OS and z/VSE environment. In a z/VM environment, MODIFY requests must be made from a logged-on user ID.

Note: The MODIFY command cannot be routed through the security network using the Command Propagation Facility (CPF).

To display the status of the site security environment, this command function has the following format:

```
TSS MODIFY STATUS
```

To enter or change control options, this command function has the following format:

```
TSS MODIFY(control_option [(suboption_list)])
```

The maximum character length of the TSS MODIFY subfield is 80 characters.

Under z/VM, if the ACID issuing the TSS MODIFY command does not have the CONSOLE attribute, you must perform one of the following actions:

- Enter the previous password of the MSCA.
- Enter an ACID/password combination for an ACID that has CONSOLE authority.

In z/VSE, CA Top Secret only processes MODIFY requests that are issued by ACIDs with the CONSOLE attribute.

In a z/OS environment, the TSS MODIFY STATUS command can be issued by any administrator type ACID or any user with USE access to TSSCMD.ADMIN.MODIFY in the CASECAUT resource class.

To display control options, administrators and users must have *one* of the following authority levels:

- CONSOLE attribute authority
- USE access to TSSCMD.ADMIN.MODIFY in the CASECAUT resource class for issuing the following display commands:
  - MODIFY VERSION
  - MODIFY FAC(*xxxxxxxx*) where *xxxxxxxx* is the facility name
  - MODIFY FAC(*xxxxxxxx*=BYPLIST) where *xxxxxxxx* is the facility name
  - MODIFY FAC(*xxxxxxxx*=RXLTLIST) where *xxxxxxxx* is the facility name
  - MODIFY CACHE(STATUS)
  - MODIFY SECCACHE(STATUS)
  - MODIFY STATS
  - MODIFY DRC(*nnn*) where *nnn* is the detailed reason code
  - MODIFY MSG(*nnnn*) where *nnnn* is the message number
  - MODIFY RPW(LIST)

  Note: All other commands are considered alter commands and require PRIVILEG access to TSSCMD.ADMIN.MODIFY in the CASECAUT resource class. For complete descriptions of control option keywords (such as DRC, MSG, and RPW), see the *CA Top Secret Control Options Guide*.

  USE access is granted through the following command:

  ```
  TSS PERMIT(acid) CASECAUT(TSSCMD.ADMIN.MODIFY) ACCESS(USE)
  ```

  Note: USE is the default access level.

To alter control options, administrators and users must have *one* of the following authority levels:

- CONSOLE attribute authority
- PRIVILEG access to TSSCMD.ADMIN.MODIFY in the CASECAUT resource class

  PRIVILEG access is granted through the following command:

  ```
  TSS PERMIT(acid) CASECAUT(TSSCMD.ADMIN.MODIFY) ACCESS(PRIVILEG)
  ```

Example: Display the Status of a Security Environment

This example displays the status of a security environment:

```
TSS MODIFY STATUS
```

Example: Display Settings for the SECCACHE Control Option

This example displays the current settings for the SECCACHE control option:

```
TSS MODIFY SECCACHE(STATUS)
```

Example: Alter Settings for the LOG Control Option

This example alters the settings for the LOG control option:

```
TSS MODIFY LOG(INIT,SMF,SEC9,MSG)
```

MOVE Function—Move an ACID

Use the MOVE function to:

- Move an ACID from one department, division or zone to another
- Change a user, ZCA, DCA and/or VCA into an SCA
- Promote or demote the type of ACID

Administrators must have authority, via ADMIN - ACID(MAINTAIN|ALL), to move ACIDs within their scope.

This command function has the following format:

```
TSS MOVE(acid) DEPARTMENT(acid)|DIVISION(acid)|ZONE(acid)
               TYPE(acid)
```

acid

Specifies the ACID being moved.

DEPARTMENT(*acid*)|DIVISION(*acid*)|ZONE(*acid*)

Specifies the department, division or zone ACID of the *destination*. Not entering a department, division or zone makes the ACID (user, DCA, VCA, ZCA, LSCA) an SCA.

TYPE(*acid*)

Specifies the ACID type you are moving to.

Example: MOVE function

This example moves a user (ACT4T) from the Accounts Receivable Department to the Accounts Payable Department (DACTPAY):

```
TSS MOVE(ACT4T) DEPARTMENT(DACTPAY)
```

Effects of MOVE if the TYPE Keyword is Omitted

The movement of an ACID from one organization to another may change the ACID type of the ACID being moved. The following table shows how each type of ACID is changed when moved into another department, division, or zone.

ACID Type	When moved into a DEPARTMENT	When moved into a DIVISION	When moved into a ZONE	When moved with no destination
USER	Remains User	Becomes VCA	Becomes ZCA	Becomes SCA
PROFILE	Remains Profile	N/A	N/A	N/A
DCA	Remains DCA	Becomes VCA	Becomes ZCA	Becomes SCA
VCA	Becomes DCA	Remains VCA	Becomes ZCA	Becomes SCA
ZCA	Becomes DCA	Becomes VCA	Remains ZCA	Becomes SCA
LSCA*	Becomes DCA	Becomes VCA	Becomes ZCA	Becomes SCA
SCA	Becomes DCA	Becomes VCA	Becomes ZCA	N/A
DEPARTMENT	N/A	Connected ACIDs and resources are moved with the department to a new division.	N/A	Remains DEPARTMENT with no division
DIVISION	N/A	N/A	Connected ACIDs and resources are moved with the division to a new zone	Remains DIVISION with no zone

LSCA can only be moved once his scope of authority has been removed. For example:

```
TSS DEADMIN(LSCA01) SCOPE(ZONE01,LSCA02)
```

Note: The following considerations apply when performing a MOVE:

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

Effects of MOVE Using the TYPE Keyword

With the TYPE keyword, an administrator can move an ACID from its original type to a new targeted type. A DEPARTMENT or DIVISION will only be specified if the new targeted ACID type requires it. The following table details TYPE keyword usage:

TYPE Specification	Keyword Required
SCA	N/A
LSCA	N/A
ZCA	ZONE
VCA	DIVISION
DCA	DEPARTMENT
USER	DEPARTMENT

Examples: MOVE with the TYPE keyword

This example moves a DCA (DCA01) into a new financial department (FINDEPT) and demote this DCA to a TYPE(USER):

```
TSS MOVE(DCA01) DEPARTMENT(FINDEPT)
                TYPE(USER)
```

This example leaves a TYPE(USER) in the same department but promotes it to a DCA:

```
TSS MOVE(USER01)TYPE(DCA)
```

P11TOKEN Function—Manage Certificates, Public Keys, and Private Key Objects

Valid on z/OS.

Use the P11TOKEN command to:

- Create or delete a token
- Display information about objects in a token
- Connect a CA Top Secret digital certificate to an existing token
- Remove a certificate from a token
- Define a token certificate to CA Top Secret

Administrators must have:

- MISC3(PTOK) authority
- Administrators without MISC3(PTOK) authority must have:
  - UPDATE access to TSSCMD.DIGTCERT.P11TOKEN.ADD in the CASECAUT resource class to use the TOKENADD keyword
  - UPDATE access to TSSCMD.DIGTCERT.P11TOKEN.DELETE in the CASECAUT resource class to use the TOKENDEL keyword
  - UPDATE access to TSSCMD.DIGTCERT.P11TOKEN.BIND in the CASECAUT resource class to use the BIND keyword
  - UPDATE access to TSSCMD.DIGTCERT.P11TOKEN.UNBIND in the CASECAUT resource class to use the UNBIND keyword
  - UPDATE access to TSSCMD.DIGTCERT.P11TOKEN.IMPORT in the CASECAUT resource class to use the IMPORT keyword
  - USE access to TSSCMD.DIGTCERT.P11TOKEN.LIST in the CASECAUT resource class to use the TOKENLST keyword

This command function has the following format:

TSS P11TOKEN *keyword keyword*

This command function is used with the keywords TOKENADD, TOKENDEL, TOKENLST, BIND, UNBIND and IMPORT.

PERMIT Function—Permit Access to Resources

Valid on z/OS, z/VSE, and z/VM.

Use the PERMIT command function to authorize ACIDs full or restricted access to resource they do not own.

Use the PERMIT command function to control:

- Who can access a resource
- How, when, and/or at what level, the resource can be accessed
- The ACTION taken when access to a resource is attempted

Resource ownership means that the user, profile, or control ACID has an access level of ALL. It may not be desirable to grant unlimited access to individual users or profiles, CA Top Secret administrators should assign resource ownership to department or division ACIDs using the ADDTO command function.

Administrators must have the appropriate resource(XAUTH) authority, via the TSS ADMIN command function, to PERMIT access to owned resources within their administrative scope. Note that RESOURCE(XAUTH) allows administrators to PERMIT access to *all* owned resources within their administrative scope. Administrators must also have explicit authority to use each access level keyword.

Given the proper administrative authority, an CA Top Secret administrator may allow any ACID to access a resource, even if the ACID is outside of the administrator's scope. The resource, however, must be within the administrator's scope of authority.

All resources defined to the RDT can also be used with the PERMIT/REVOKE command function.

A resource must be owned before access can be permitted.

Resources may not be permitted to group, zone, department, or division ACIDs.

This command function has the following format:

```
TSS PERMIT(acid) keyword(p-fix)
                 ACCESS(level)
                 keyword(oper)
```

ACID

Specifies the ACID of user or job for whom access is being permitted.

Keyword

Specifies the keyword for type of resource to which access is being permitted. For example DSNAME and VOLUME

p-fix

Specifies the prefix or resource name. A specific level of ACCESS to the resource, if applicable; if no entry is made, CA Top Secret usually assigns a default access level based on the resource type. For example, the default for data set is READ.

Level

Specifies the manner in which a resource can be used once accessed. For example NONE, READ, and WRITE.

Keyword

Additional access keywords and their associated options. For example: DAYS(WEEKENDS), LIBRARY(SYS2.TESTLIB), and ACTION(FAIL)

Multiple PERMITs

CA Top Secret allows the use of prefixing and masking for resource identification. Therefore, an administrator can enter multiple permissions to the same resource for the same user. A REVOKE command can remove a single or multiple permissions. CA Top Secret uses a security validation algorithm to determine whether a resource access request should be granted or denied. For information on this algorithm, see the *User Guide.*

PERMIT Applicable Keywords

This command function uses the keywords:

- ACID
- ACTION
- APPLDATA
- CALENDAR
- DAYS
- FACILITY
- FOR
- IMSMSC
- LIBRARY
- MAPREC
- MASKREC
- MODE
- PRIVPGM

- SELECT
- TIMEREC
- TIMES
- UNTIL
- VMUSER

Duplicate Permissions

No duplicate permissions are stored in the CA Top Secret Security File. If a permission is issued to a user or profile that exactly matches an existing permission, CA Top Secret will issue PERMIT FUNCTION SUCCESSFUL. No error messages are issued since the administrator requested that the permit be stored with the user or profile, and CA Top Secret verified that this request was accomplished.

The matching criteria require that all fields coded on the new permit exactly match the existing permission. If the new permit or existing permission has extra parameters, this would not constitute a match and the new permit would be stored on the Security File.

Best Match

CA Top Secret searches a user's entire Security Record before making an access determination. This determination, in part, is based on the permission that contains a resource name or prefix that is the “best match” for the resource name or prefix being requested. This best match is based on the length of the prefix. If USER01 requests access to data set 'AB.CD' and CA Top Secret encounters the following entries during its security validation search:

```
TSS PERMIT(USER01) DSNAME(AB.C)
                   ACCESS(UPDATE)

TSS PERMIT(USER01) DSNAME(AB)
```

Then CA Top Secret will grant access to the PERMIT containing AB.C since this data set prefix is the best match for the requested 'AB.CD'.

Equal Prefix Lengths

If CA Top Secret encounters two matching PERMITs with equal prefix lengths while searching a Security Record, the access determination is based on the following process:

- The first PERMIT containing a matching prefix is considered the current best match. If no other PERMITS in the Security Record contain a matching prefix, CA Top Secret will base its access decision on this PERMIT command function.
- If the PERMIT in above *authorizes* access, contains the ACCESS(NONE) restriction, or has the ACTION(DENY) attribute, CA Top Secret ignores all subsequent PERMITs that contain resource prefixes of equal length. CA Top Secret will, however, examine PERMITs containing matching prefixes of greater length.
- If the PERMIT found in the first step does not allow access, does not contain the ACCESS(NONE) restriction, or does not have the ACTION(DENY) attribute, then subsequent PERMITs of equal prefix lengths are searched.

Selectively Revoking PERMITs

You can issue a specific REVOKE to remove one permission by listing the user/profiles XAUTH data, and then issuing the REVOKE exactly as it appears in the listing. For example, the output of the following TSS LIST command would enable the authorization following it to be revoked:

```
XA DATASET = SYS1.PROCLIB
   ACCESS  = READ
   PRIVPGM = IEBCOPY

TSS REVOKE(user) DSNAME(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.

Everything that can be coded on a PERMIT can be coded on a REVOKE with the exception of the FOR parameter. In this situation, the FOR parameter will list as the UNTIL parameter. Therefore, only the UNTIL parameter should be used with the TSS REVOKE command function.

If you code the REVOKE with only the resource and no other parameters, then all matches of the resource are revoked.

Example: revoke permission

This example revokes all permissions that match it.

```
TSS REVOKE(user) DSNAME(SYS1.PROCLIB)
```

Using the GENERIC - NONGENERIC Attribute

The RDT attribute, NONGENERIC, causes a general resource to be treated as a fully qualified name rather than as a generic prefix.

Note: The GENERIC/NONGENERIC attribute affects security resource checks, it has no effect on CICS Bypass List processing.

The NONGENERIC attribute can support both long and short resource classes. This attribute does not, however, support the resources that support masking characters.

For example, an administrator can permit a user to resource OTRAN(PSRV), which would allow access to PSRV as well as PSRVTEST or any other OTRAN whose first four characters match the prefix, PSRV. If the NONGENERIC attribute is activated, a permit to OTRAN(PSRV), however, only allows the user to execute transaction PSRV and not PSRVTEST. In order for the user to be allowed to issue either transaction, the permit must be done to OTRAN(PSRV(G)).

To alter a particular general resource class to conform to the non-generic attribute, enter:

```
TSS REPLACE(RDT) RESCLASS(OTRAN)
                 ATTR(NONGENERIC)
```

The OTRAN keyword is the resource class to be altered.

To remove the NONGENERIC attribute, enter:

```
TSS REPLACE(RDT) RESCLASS(OTRAN)
                 ATTR(GENERIC)
```

When changing the resource class from GENERIC to NONGENERIC, or from NONGENERIC to GENERIC, the security validation behavior for all existing definitions is preserved. However, resources will list differently and administrative commands which follow will have a different effect.

For example, if an attribute of a resource class is changed from GENERIC to NONGENERIC, any resource permitted prior to the change displays a (G) to indicate the cross-authorization remains GENERIC.

Also, attempts to revoke such a permit may fail, with either TSS0384E: “RESOURCE NOT FOUND IN SECURITY RECORD”, (if the (G) is not included on the REVOKE statement), or TSS0244E: “INVALID SUBFIELD LENGTH FOR KEYWORD - keyword” (if the (G) is included).

In this case, the attribute of the resource class should be temporarily changed back from NONGENERIC to GENERIC (when no other administration is taking place). The REVOKE (without the (G)) should then succeed, and the resource attribute may again be changed to NONGENERIC.

The NONGENERIC attribute applies to the following resources *by default:*

- IUCV
- VMCF
- VMDIAL
- VMMACH
- VMRDR

Mask and Prefix for Other Resources

Resource masking allows administrators to group resources with similar characteristics. Special characters are specified to represent variations between resource names. Prefixing allows administrators to group similar resources defined by CA Top Secret simultaneously.

Note: Resource classes may also be administered to support masking characters by manipulating the RDT MASK/NOMASK attribute.

An administrator can PERMIT access to all owned resources within a NOMASK resource class by using the special resource name:

```
*ALL*
```

For resource classes with the MASK attribute, a similar PERMIT can be managed with the resource name:

```
*****
```

Adding this resource is a formality and has no effect on resources not already protected. Its main importance is with PERMIT because it allows the administrator to allow access to all owned resources of a RESCLASS. This is a different concept than applying the DEFPROT attribute to a resource class. When DEFPROT is applied, all resources are implicitly protected without the necessity of formal ownership; in order to grant access under DEFPROT, the administrator must still own and permit the resources (which may include *ALL*).

Data Set Prefixing

Prefixing allows the CA Top Secret administrator to group a set of similar data sets together, and define them by a 2 to 26 character prefix. For example, all data sets used by the Publications Department could be prefixed by TECHPUBS. Then, the administrator could PERMIT a user to access any data set prefixed by TECHPUBS.

```
TSS PERMIT(USER01) DSNAME(TECHPUBS.)
```

This eliminates the need to permit access to 'TECHPUBS.PROD.SCEDS' and 'TECHPUBS.GRAPHICS.SCEDS'.

A prefix may span several data sets name index levels. For example, DSNAME(TECHPUBS.PR).

Resource Masking

Resource masking is another method of reducing the number of resource definitions required to implement widespread protection. Masking allows administrators to group resources with similar characteristics, and use special characters to represent variations between resource names.

Floating "-"

The character "-" represents a variable number of characters. The following table provides an example of floating:

Entry:	Matches:	But Not:
ACCT-VEND	ACCTPAY.VENDOR	ACC.VEND
	ACCTVEND	AP.ACC.VEND

In this case, the 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 prefix ACC.VEND does not begin with ACCT and therefore does not match the entry.

Floating characters can cross node (also called qualifier or index) boundaries. Floating characters cannot be used with other mask types.

Variable "*"

The asterisk “*” can be used to represent zero to eight characters. The following table details variable usage:

Entry:	Matches:	But Not:
ACCT*M.DATA	ACCTPAYM.DATA	ACCTPAYD.DATA
*LAB	LAB	NEW.JERSEY.LAB
*RAT	TESTRAT.LAB	TRAP.ALL.RATS

In the first entry, the asterisk 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 does not match NEW.JERSEY.LAB since NEW.JERSEY is more than eight characters long. Multiple asterisks can be listed in series to equal up to 44 characters; therefore, **LAB would match NEW.JERSEY.LAB.

Index ".*."

The characters “.*.” or “*.” appearing at the beginning of the data set name are used to represent a one to eight character index. The following table details index usage:

Entry:	Matches:	But Not:
*.BALL	BASKET.BALL	BALL.GAME
CICS.*.*.F	CICS.RUM.TSS.FIL	CICS.FEATURES

The index mask is an extension of the variable mask that allows the CA Top Secret administrator to specify where index levels must appear in a data set name. The first entry specifies that .BALL must be prefixed by one to eight characters; therefore, the data set BALL.GAME does not match this definition. The second entry specifies that two indexes must appear between CICS. and .F. CICS.FEATURES contains no indexes between CICS. and .F and therefore does not match the mask.

Fixed Position "+"

The character “+” is used to represent any single character within a data set name. The following table includes examples of the fixed position "+" usage:

Entry:	Matches:	But Not:
A123+.TSO	A1234.TSO	A123.TSO
ACC+VEND	ACCTVEND	AP.ACC.VEND

ACID "%"

The character “%” represents the ACID of the user specified in the TSS command.

If TSS PERMIT(USER52) DSNAME(%.MAIL) ACCESS(ALL), then USER52 will have full access to the data set named USER52.MAIL.

This capability can be used to permit all TSO users full access to data sets prefixed by their ACID.

The MSCA must own the "%." character. Only the MSCA can own a special character.

```
TSS ADDTO(MSCA acid) DSNAME(%.)
```

An CA Top Secret Administrator may now PERMIT all TSO users to have all access to any data set prefixed by an ACID.

```
TSS PERMIT(ALL) DSNAME(%.)
                FACILITY(TSO)
                ACCESS(ALL)
```

CA Top Secret recognizes a special technique to specify a portion of the userid used as a mask. The special technique uses the % character in conjunction with values identifying the start and length of the userid being entered.

Example: using the %

This example permits TUSER01 access to data set USER01:

```
TSS PERMIT(TUSER01) DSNAME(%26%)
```

Mask Combinations

All masking characters, *except* the floating mask, may be combined. The following table provides an example of how you can and cannot combine masks:

Entry:	Matches:	But Not:
MAR++.*.SUM	MAR86.A.SUM	MARCH.SUM

Illegal Mask Combinations

An example of an illegal combination is:

```
CSSY.*-83     BOTL-WORK+++.COMP
```

Floating characters “-” may not be combined with other masking characters.

Minidisk Prefixing

Prefixing allows the CA Top Secret administrator to group a set of similar minidisks together, and define them by one generic prefix. For example, all of user MAINT's 19x minidisks (like 0190, 019D, 019E) could be grouped together using a generic prefix.

Minidisk Masking

Masking (or “patterning”) is another method of reducing the number of definitions required to implement widespread minidisk protection. Masking allows administrators to group minidisks with similar characteristics, and use special characters to represent variations between minidisks.

Variable "*"

The asterisk (*) can be used to represent zero to eight characters. The following table provides an example of asterisk usage:

Entry:	Matches:	But Not:
PAY*.019	PAYUSER1.0191	PAYDEPT.0391
	PAY.0191	PRODUCTS.0191
	PAY03.0192	PAY.0091

In the first entry, the asterisk is used to represent any 0 to 8 characters between PAY and (.); thus the entry matches PAYUSER1.0191 but not PAYDEPT.0391. The characters (.019) are a simple prefix; the first three characters of the device address or device number must match them exactly.

Fixed Position "+"

The character “+” is used to represent any single character within a minidisk name. The following table provides an example of fixed position "+" usage:

Entry:	Matches:	But Not:
PAY+++.0191	PAYROL.0191	PAYDEPT.0191

ACID "%"

The character “%” represents the ACID of the user requesting access.

If TSS PERMIT(USER01) VMMDISK(%019), then USER01 will have full access to minidisks USER01.0191, USER01.019E, and so on.

This capability can be used to permit all VM ACIDs access to minidisks prefixed by their own ACID.

The MSCA must own the "%." character. Note that only the MSCA can own this special character.

```
TSS ADDTO(MSCA acid) VMMDISK(%.)
```

An CA Top Secret administrator may now PERMIT all VM users to have ALL access to any minidisk prefixed by their ACID.

```
TSS PERMIT(ALL) VMMDISK(%.)
                FACILITY(VM)
                ACCESS(ALL)
```

Combinations

Mask characters may be combined with the exception of a dash (-). The following table provides an example of combinations

Entry:	Matches:	But Not:
SUP*1.+91	SUPMAH1.191	SUPMAH.191

REFRESH Function—Renew ACIDs

Valid on z/OS, z/VSE, and z/VM.

Use the REFRESH command function to renew the ACIDs in any address space in the security environment. Use this command in multi-user address spaces like CICS and IMS, where an ACID can have multiple instances of signed on users.

Issuing the REFRESH command against a target region refreshes all occurrences of an ACID. As a result, the user does not need to log off and log back on for administrative changes to take effect.

All users are allowed to REFRESH their own security environment.

The ACID being refreshed must be within the administrator's scope.

This command function has the format:

```
TSS REFRESH[(acid) [JOBNAME(job|*)]]
```

no operands (blank)

Refreshes one's own security environment.

acid

Refreshes all occurrences of a specified ACID in the address space where the command is issued

JOBNAME

Refreshes one of the following:

job

Refreshes all occurrences of a specified ACID in the address space of a given jobname.

*****

Refreshes all occurrences of a specified ACID in all address spaces.

Example: REFRESH function

This example gets a new copy of the signed on user's ACID in the current address space:

```
TSS REFRESH
```

REKEY Function—Create Certificate from Existing Certificate

Valid on z/OS.

Use the REKEY command function to create a new certificate from an existing certificate with a new public/private key pair. The REKEY command is the first step of a rekey-rollover process to retire the use of an existing private key.

The REKEY command copies the subject's distinguished name, key usage and subject alternate name from the existing certificate. The new certificate is self-signed and saved under the same logonid or CERTAUTH or CERTSITE.

If the new certificate needs to be signed by a third party CA or CA Top Secret, issue a TSS GENREQ command to copy the new certificate to a dataset then FTP the new certificate to the third party CA or input to the TSS GENCERT so it may be signed with CA Top Secret. Do this prior to the TSS ROLLOVER command.

Specify a DIGICERT name as part of all REKEY functions since the DIGICERT keyword indicates the name used in the digital certificate.

Administrators must have:

- ACID(MAINTAIN) and MISC4(CERTGEN) for users within their scope
- MISC4(CERTSITE) for CERTSITE ACID
- MISC4(CERTAUTH) for CERTAUTH ACID
- Administrators without the above authorities can issue the REKEY command if they have:
  - UPDATE access to TSSCMD.CERTUSER.REKEY in the CASECAUT resource class when the certificate is associated with a user ACID
  - UPDATE access to TSSCMD.CERTSITE.REKEY in the CASECAUT resource class when referencinging a site certificate
  - UPDATE access to TSSCMD.CERTAUTH.REKEY in the CASECAUT resource class when referencinging a certificate-authority certificate

This command function has the following format:

```
TSS REKEY {acid|CERTAUTH|CERTSITE}
          [DIGICERT(existing–certificate–id)]
          [NEWDIGIC(new–certificate–id)]
          [NEWLABLC(new–certificate–label)]
          [KEYSIZE(nnnn)]
          [ICSF|PCICC|NISTECC|BPECC]
          [NBDATE({not–before–date} NBTIME(not–before–time)]
          [NADATE(not–after=date|) NATIME(not–after–time)]
          [LABLPKDS]
```

DIGICERT(*id*)

(Mandatory) Specifies a case sensitive character ID that identifies existing certificate.

Range: 1 to 8 characters

NEWDIGIC(*id*)

(Mandatory) Specifies a case sensitive character ID of the new certificate.

NEWLABLC(*label*)

(Optional) Specifies the new certificate's a character label. The label can contain blanks and mixed case characters. The new label must be unique to the logonid with which the new certificate is associated. If a label is not specified, the label field defaults to the upper case version of the ACID.

Note: For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank”s Certificate. If a single apostrophe is specified in the Label value, the value is considered invalid.

Range: 1 to 32 characters

KEYSIZE

- Specifies the size of the private encryption key in decimal bits.
- **Limits:**
  - Non-NISTECC and non-BPECC: any numeric value from 512 to 4096.
  - For NISTECC, one of the following: 192, 224, 256, 384, 521.
  - For BPECC, one of the following: 160, 192, 224, 256, 320, 384, 512.
- **Default:**
  - Non-NISTECC and non-BPECC:1024.
  - NISTECC and BPECC: 192.

The maximum key size is dependent on the private key type.

Private key type maximum key sizes are:

- BPECC key—512 bits
- NISTECC key—521 bits
- ICSF RSA key—1024 bits
- DSA key —2048 bits
- Non-ICSF RSA key—4096 bits
- PCI-class cryptographic coprocessor RSA key—4096 bits

Shorter ECC keys have key strengths comparable to longer RSA keys. The following table displays the comparable strength of each key type:

RSA Key Size (in bits)	NISTECC Key Size (in bits)	BPECC Key Size (in bits)
1024	192	160 or 192
2048	224	224
3072	256	256 or 320
7680	384	384
15360	521	512

Currently, the standard key sizes for RSA keys are as follows:

- 512—Specifies a low-strength key
- 1024—Specifies a medium-strength key
- 2048—Specifies a high-strength key
- 4096—Specifies a very high-strength key

ICSF

(Optional) Indicates that the generated private key is placed in ICSF. If ICSF, PCICC, or LABLPKDS is not specified with ADD, the key is stored in the security file as a non-ICSF key. If the DSN parameter was also specified and an existing certificate is replaced, the existing certificate is also placed in ICSF. If ICSF is not active and configured for PKA operations, an error message is displayed when attempting to insert or use the private key.

PCICC

(Optional) Specifies that the key pair should be generated using the PCI Cryptographic Coprocessor and that the private key should be stored in ICSF. When PCICC is not specified, the key pair is generated using software. PCICC cannot be used with the DSA, DSN, or ICSF parameters. If a PCI cryptographic coprocessor is not present or operational, or if ICSF is not active or configured for PKA operations, an error message is displayed and processing will terminate.

NISTECC

(Optional) Specifies the key pair should be generated using National Institute of Standards and Technology (NIST) algorithm instead of the RSA algorithm. This parameter cannot be used with the ICSF, DSA, or BPECC parameters.

BPECC

(Optional) Specifies to generate the key pair using the brainpool ECC algorithm instead of the RSA algorithm. This parameter cannot be used with the ICSF, DSA, or NISTECC parameters.

Notes:

- If you do not specify ICSF, PCICC, NISTECC, or BPECC, the key pair is generated using software and stored in the CA Top Secret database.
- REKEY processing prevents the downgrade from an ICSF/PCICC private key to a private key stored in the security product database.
- If you do not specify ICSF, PCICC, NISTECC, or BPECC, REKEY will use what was used on the original certificate.
- REKEYing from a NISTECC or BPECC certificate to a non-ECC certificate (and conversely) is not allowed.
- You do not need to specify anything to go from NISTECC to NISTECC or from BPECC to BPECC.
- Specify NISTECC to go from BPECC to NISTECC (and conversely).

NBDATE/NBTIME Format(*mm/dd/yy*) Time(*hh:mm:ss*)

Indicates the date and time that the certificate becomes active. If no expire date is specified, the active year specified must be before 2048, because the expire date defaults to the active day and time plus one year.

Range: 1950 to 2049

Time Default: 000000

Date Default: Current day and time

NADATE/NATIME Format(*mm/dd/yy*) Time(*hh:mm:ss*)

(Optional) Indicates the date and time that the certificate expires.

Range: 1950 to 2049

Time Default: 000000

Date Default: The active day and time plus one year.

LABLPKDS

(Optional) Specifies the PKDS label of the record created in the ICSF Public Key Data Set (PKDS). This field is used with the ICSF, PCICC, NISTECC, and BPECC keywords. If LABLPKDS is specified without ICSF or PCICC, the key will be generated by the hardware and saved in CRT format in the ICSF PKDS. If NISTECC or BPECC is specified, the key will be an ECC key. Otherwise, the key that is generated will be an RSA key.

Specify a value of (*) to take the value from the LABLCERT keyword. In this case, LABLCERT must be specified along side LABLPKDS(*). If LABLPKDS(*) is specified without the LABLCERT keyword, an error message is displayed.

The PKDS label must conform to ICSF label syntax rules. The first character must be alphabetic or national.

Valid characters: Alphanumeric, national (@,#,$) or period(.).

Range: Up to 64 characters

Example: REKEY function

This example creates the new certificate Locca4 for the existing certificate CERTAUTH.

```
TSS REKEY(CERTAUTH) DIGICERT(Locca4)
                    NEWDIGIC(locca5)
                    NADATE(12/31/08)
```

REMOVE Function—Remove Resource Ownership or Resources

Valid on z/OS, z/VSE, and z/VM.

Use the REMOVE command function to remove ownership of the keyword specified in the ADDTO function. Ownership cannot be removed until permitted access is revoked, if applicable. REMOVE is the functional opposite of ADDTO.

Issuing the REMOVE command to remove a profile for a signed-on ACID automatically triggers a type 71 RACF Event Notifications (ENF) signal, which provides notification about the change to the ACID's security record. Applications that receive the signal can take action (for example, immediately refreshing the ACID's security record in a CICS remote region).

This command function has the following format:

```
TSS REMOVE(acid) keyword[(operand)]
```

acid

Specifies the ACID from which the resource or attribute ownership is being removed.

keyword

Specifies the type of resource or attribute being removed.

operand

Specifies the specific prefix, name, or value of the resource or attribute. When the keyword is assigned with a unique operand value, the operand value can be omitted. The parentheses for the operand are still required.

Avoid mixing unrelated resources, fields, and keywords in the same ADD/REMOVE command for a user, to avoid ambiguities that might not be anticipated.

Example: Remove Keywords Explicitly

This example removes keywords explicitly:

```
TSS REMOVE(user1) NOLCFCHK
                  TSOLPROC(proc396)
                  TSOJCLASS(a)
```

Example: Remove Keywords Implicitly

This example removes the keywords implicitly:

```
TSS REMOVE(user1) NOLCFCHK
                  TSOLPROC()
                  TSOJCLASS()
```

Example: Specify a Resource Subfield when Removing a Resource from a User

When removing a resource from a user, the resource subfield must be specified. For example, the data set name added with the following command can only be removed explicitly:

```
TSS ADD(user1) DSNAME(xxxxxxx.yyyyyyy.zz)
```

The following specification removes the resource:

```
TSS REMOVE(user1) DSNAME(xxxxxxx.yyyyyyy.zz)
```

The following specification does *not* remove the resource:

```
TSS REMOVE(user1) DSNAME()
```

More information:

ADDTO Function—Add Resource Ownership and Attributes (see page 37)

RENAME Function—Change ID Assigned to ACID

Valid on z/OS, z/VSE, and z/VM.

Use the RENAME command function to change the access ID of any ACID. An ACID cannot be renamed if it is currently permitted to another ACID.

Administrators must have authority, via the ADMIN - ACID(MAINTAIN | ALL) function, to rename an ACID within their scope.

This command function has the following format:

```
TSS RENAME(acid) ACID(new acid)
```

Acid

Specifies the ACID to be renamed.

new acid

Specifies the new ACID. The only characters used for an ACID are:

- Alphabetical: A-Z
- Numeric: 0-9
- National: $ # @ % & = ?

Other characters may cause unpredictable results and are not allowed.

Example: RENAME function

This example changes the ACID TSSREC to BIGBOSS:

```
TSS RENAME(TSSREC) ACID(BIGBOSS)
```

Global Records

The ALL, AUDIT, STC, NDT, RDT, and DLF records cannot be renamed.

For example, CA Top Secret will *not* accept the entry:

```
TSS RENAME(ALL) ACID(GLOBAL)
```

RENEW Function—Renew a Digital Certificate

Valid on z/OS.

The RENEW command function renews a digital certificate. Use the RENEW command if a certificate is expiring and you want to continue to use the certificate. The RENEW command copies the information from an existing certificate and applies it to a certificate. This command can only renew certificates created by CA Top Secret, it cannot renew certificates from external certificate authorities. Certificates from external certificate authorities must be renewed manually using the PKCS 10 data set.

Notes:

- Because the DIGICERT keyword indicates the name used in the digital certificate, specify a DIGICERT name as part of all RENEW functions.
- You can enter additional keywords to modify the renewing certificate.
- Use the NADATE keyword to specify an expiration date. If no date is specified, the certificate expires one year from the date that the certificate is renewed.
- The keywords ICSF, PCICC, and LABLPKDS are only valid if the original certificate is NON-ICSF.
- Before you renew a certificate, we recommend as a best practice that you export the certificate to a data set to save it, either using the RENEW command or manually.

Administrators must have:

- ACID(MAINTAIN) and MISC4(CERTGEN) for users within their scope
- MISC4(CERTSITE) for CERTSITE ACID
- MISC4(CERTAUTH) for CERTAUTH ACID

This command function has the following format:

```
TSS RENEW {(CERTAUTH|CERTSITE|acid)}
    DIGICERT(8-byte-name)
    [SUBJECTN('CN="common-name" T="title" OU="org-unit-name1,org-unit-name2"
    O="organizational-name" L="locality" ST="state-or-province" C="2–digit–only
    country code"')]
    [NBDATE(mm/dd/yy) NBTIME(hh:mm:ss)]
    [NADATE(mm/dd/yy) NATIME(hh:mm:ss)]
    [LABLCERT(label name)]
    [ICSF|PCICC]
    [SIGNWITH(acid,digicert)]
    [KEYUSAGE(HANDSHAKE DATAENCRYPT DOCSIGN CERTSIGN)]
    [ALTNAME('IP=numeric-IP-address DOMAIN=internet-domain-name
    EMAIL=email-address URI=universal-resource-identifier')]
    [LABLPKDS]
```

CERTAUTH|CERTSITE|acid

Specifies the acid that will use the certificate.

CERTAUTH

Specifies the certificate as a certificate-authority certificate.

CERTSITE

Specifies the certificate as a site certificate.

acid

Specifies the user associated with the certificate.

DIGICERT(8-byte-name)

Specifies the keyword that identifies the digital certificate being renewed.

SUBJECTN('CN="common-name" **T=**"title" **OU=**"org-unit-name1,org-unit-name2" **O=**"organizational-name" **L=**"locality" **ST=**"state-or-province" **C=**"2—digit—only country code"**')**

(Optional) Specifies the distinguished name of the ACID. When you specify multiple parameters for SUBJECTN, surround the parameter list with single quotation marks. You can specify multiple values for OU=.

Default: The name field of the ACID

NBDATE/NBTIME Format(*mm/dd/yy*) Time(*hh:mm:ss*)

Indicates the date and time that the certificate becomes active. If no expire date is specified, the active year specified must be before 2048, because the expire date defaults to the active day and time plus one year.

Range: 1950 to 2049

Time Default: 000000

Date Default: Current day and time

NADATE/NATIME Format(*mm/dd/yy*) Time(*hh:mm:ss*)

(Optional) Indicates the date and time that the certificate expires.

Range: 1950 to 2049

Time Default: 000000

Date Default: The active day and time plus one year.

LABLCERT(*label-name*)

(Optional) Defines the label name of the certificate being renewed.

ICSF

(Optional) Indicates that the generated private key is placed in ICSF. If ICSF, PCICC, or LABLPKDS is not specified with ADD, the key is stored in the security file as a non-ICSF key. If the DSN parameter was also specified and an existing certificate is replaced, the existing certificate is also placed in ICSF. If ICSF is not active and configured for PKA operations, an error message is displayed when attempting to insert or use the private key.

PCICC

(Optional) Specifies that the key pair is generated using the PCI Cryptographic Coprocessor and that the private key is stored in ICSF. When PCICC is not specified, the key pair is generated using software. PCICC cannot be used with the DSA, DSN, or ICSF parameters.

If a PCI cryptographic coprocessor is not present or operational or if ICSF is not active or configured for PKA operations, an error message is displayed and processing terminates. If ICSF, PCICC, or LABLPKDS is not specified, the key pair is generated using software and stored in the security file as a non-ICSF key.

SIGNWITH(acid,digicert)

(Optional) Specifies the digital certificate signing the certificate. If not specified, the certificate is signed with the private key of the certificate being generated, creating a self-signed certificate.

KEYUSAGE(HANDSHAKE DATAENCRYPT DOCSIGN CERTSIGN)

(Optional) Specifies the appropriate values for the KeyUsage certificate extension. If the KEYUSAGE data contains more than one value, place single quotation marks around the data.

Example: KEYUSAGE('HANDSHAKE DATAENCRYPT')

HANDSHAKE

(Optional) Facilitates identification and key exchange during security handshakes, such as SSL, which set the digitalSignature and keyEncipherment indicators. When the key pair is generated using the DSA algorithm, only the digitalSignature bit is set because the keys cannot be used for encryption.

DATAENCRYPT

(Optional) Encrypts data, which sets the dataEncipherment indicator. When the key pair is generated using the DSA algorithm, you cannot use the DATAENCRYPT keyword in the Keyusage parameter.

DOCSIGN

(Optional) Specifies a legally binding signature, which sets the nonRepudiation indicator.

CERTSIGN

(Optional) Specifies a signature for other digital certificates and CRLs, which sets the keyCertSign and cRLSign indicators.

ALTNAME ('IP=numeric-IP-address **DOMAIN=**internet-domain-name **EMAIL=**email-address **URI=**universal-resource-identifier**')**

(Optional) Specifies the appropriate values for the SubjectAltname extension. When you specify multiple parameters for ALTNAME, surround the parameter list with single quotation marks. Separate multiple parameters by a space.

Example: ALTNAME('IP=200.100.10.1 EMAIL=my.email@test.net')

LABLPKDS

(Optional) Specifies the PKDS label of the record created in the ICSF Public Key Data Set (PKDS). This field is used with the ICSF, PCICC, NISTECC, and BPECC keywords. If LABLPKDS is specified without ICSF or PCICC, the key is generated by the hardware and saved in CRT format in the ICSF PKDS. If NISTECC or BPECC is specified, the key is an ECC key; otherwise, the key is an RSA key.

Note: The PKDS label must conform to ICSF label syntax rules. The first character must be alphabetic or national.

To take the value from the LABLCERT keyword, include a LABLCERT specification in your syntax and specify the following syntax for LABLPKDS:

```
LABLPKDS(*)
```

Valid characters: Alphanumeric, national (@,#,$), or period(.).

Range: Up to 64 characters

Example: RENEW function

This example creates new certificate Locca4 for the existing certificate CERTAUTH.

```
TSS RENEW(CERTAUTH) DIGICERT(Locca4) NADATE(12/31/11)
```

REPLACE Function—Change ACIDs Attributes

Valid on z/OS, z/VSE, and z/VM.

Use the REPLACE command function to change the values, names, or data of CA Top Secret attributes or keywords assigned to ACIDs.

Full use of the REPLACE function requires a combination of administrative authorities. See the specific attribute or keyword in the detailed reference pages for the appropriate administrative authority.

Administrators can use the REPLACE function only to maintain ACIDs under their administrative scope.

The command has the following format:

```
TSS REPLACE(acid) attribute keyword(value)
```

Acid

Specifies the ACID affected by the REPLACE function.

attribute

Specifies the attribute or keyword changed by the REPLACE function.

value

Specifies the new attribute or keyword's value, name, or data that REPLACE the current information.

REPLACE Applicable Keywords

This command function uses the keywords:

ACLST, ATTR, CERTNSER, CONVSEC, DEFNODES, DEFACC, DFLTSLBL, DIGICERT, FACILITY, FOR, GID, IMSMSC, INSTDATA, ISSUERDN, KERBVIO, LABLCERT, LANGUAGE, LTIME, MASTFAC,MCSALTG, MCSAUTH, MCSAUTO, MCSCMDS, MCSDOM, MCSKEY, MCSLEVL, MCSLOGC, MCSMFRM, MCSMGID,MCSMON, MCSROUT MCSSTOR, MCSUD, NAME, OPCLASS, OIDCARD, OPIDENT, OPPRTY, PASSWORD, PKCSPASS, PHYSKEY, PSTKAPPL, RESCLASS, SCTYKEY, SERIALNUM, SESSKEY, SESSLOCK, SITRAN, SMSAPPL, , SMSDATA, SMSMGMST, SMSSTOR, TARGET, TSOCOMMAND, TSODEFPRFG, TSODEST, TSOHCLASS, TSOJCLASS, TSOLACCT, TSOLPROCTSOLSIZE, TSOMCLASS, TSOMSIZE, TSOOPT, TSOSCLASS, TSOUDATA, TSOUNIT, TZONE, WAACCNT, WAADDR1, WAADDR2, WAADDR3, WAADDR4, WABLDG, WAADEPT, WAIT, WANAME, and WAROOM.

All resources defined to the RDT can also be used with the REPLACE command function.

Example: REPLACE function

This example changes a user's password:

```
TSS REPLACE(USER77) PASSWORD(QUACK)
```

REVOKE Function—Revoke Resource Access

Valid on z/OS, z/VSE, and z/VM.

Use the REVOKE command function to revoke access to ownable resources when no longer needed, or when the access restrictions (levels and/or controls) must be changed. A command can revoke multiple permissions or one specific permission.

Administrators must have the appropriate resource(XAUTH) authority, via the TSS ADMIN command function, to revoke access to owned resources within their administrative scope. Note that RESOURCE(XAUTH) allows administrators to revoke access to *all* owned resources within their administrative scope. Administrators must also have explicit authority to use each access level keyword.

Given the proper administrative authority, an CA Top Secret administrator may allow *any* ACID to access a resource, even if the ACID is outside of the administrator's scope. The resource, however, must be within the administrator's scope of authority.

All resources defined to the RDT can also be used with the REVOKE command function.

This command function has the following format:

```
TSS REVOKE (acid) keyword(p-fix)
                  ACCESS(level)
                  keyword(oper)
```

ACID

Specifies the ACID of the user or job access is being revoked from

Keyword

Specifies the keyword for the type of resource access is being revoked for. For example, DSNAME and VOLUME.

p-fix

Specifies the prefix or resource name. A specific level of ACCESS to the resource, if applicable; if no entry is made, CA Top Secret usually assigns a default access level based on the resource type. For example, the default for data set is READ

Level

Specifies the manner in which a resource can be used once accessed. For example, NONE, READ, and WRITE.

Keyword

Specifies the additional access keywords and their associated options. For example, DAYS(WEEKENDS), LIBRARY(SYS2.TESTLIB), and ACTION(FAIL).

REVOKE Applicable Keywords

This command function uses the keywords:

ACID, ACTION, APPLDATA, CALENDAR, DAYS, FACILITY,
FOR, IMSMSC, LIBRARY, MAPREC, MASKREC, MODE
PRIVPGM, SELECT, TIMEREC, TIMES, UNTIL, and VMUSER.

ROLLOVER Function—Replace Original Certificate

Valid on z/OS.

Use the ROLLOVER command function to specify the original certificate superseded by the new certificate. The ROLLOVER command is the final step in the process to create a new certificate from an existing certificate with a new public/private key pair.

The ROLLOVER command function performs the following activities:

- Deletes the private key of the original certificate so that it can no longer be used to sign or encrypt
- Replaces the original certificate with the new certificate in every key ring to which the old certificate is connected
- Copies the serial number base from the original certificate to the new certificate

When the rollover is complete, the new certificate is used as if it were the original certificate. The original certificate is still available to verify signatures and decrypt data but can no longer be used to sign or encrypt.

Administrators must have:

- ACID(MAINTAIN) and MISC4(CERTGEN) authority for users within their scope
- MISC4(CERTSITE) authority for CERTSITE ACID
- MISC4(CERTAUTH) authority for CERTAUTH ACID

If you do not have administrative authority, you can issue the ROLLOVER command if you have the following authorities in the CASECAUT resource class:

- UPDATE access to TSSCMD.CERTUSER.ROLLOVER in the CASECAUT resource class when the certificate is associated with a user ACID
- UPDATE access to TSSCMD.CERTSITE.ROLLOVER in the CASECAUT resource class when referencing a site certificate
- UPDATE access to TSSCMD.CERTAUTH.ROLLOVER in the CASECAUT resource class when referencing a certificate-authority certificate

This command function has the following format:

```
TSS ROllOVER     {acid|CERTAUTH|CERTSITE|}
                 DIGICERT(old_certificate_id)
                 NEWDIGIC(new_certificate_id)
                 [FORCE]
```

acid

Designates the user ACID associated with the certificate.

CERTAUTH

Designates the certificate as a certificate-authority certificate.

CERTSITE

Designates the certificate as a site certificate.

DIGICERT(*old_certificate_id*)

(Mandatory with ROLLOVER keyword) Specifies a case-sensitive character ID (original certificate) that identifies the certificate with the user ACID.

Range: 1 to 8 characters

NEWDIGIC(*new_certificate_id*)

(Mandatory with ROLLOVER keyword) Specifies a case-sensitive character ID that identifies the new certificate.

Range: 1 to 8 characters

FORCE

Performs the rollover unconditionally (without conducting any security checks).

Important! If DIGICERT and NEWDIGIC have the same value, and you have also specified the FORCE keyword, the product deletes the private key of the certificate.

Example: Replace the TEN Certificate with the NINE Certificate

This example completes the re-keying of the TEN certificate by replacing the TEN certificate with the NINE certificate:

```
TSS ROLLOVER(CERTSITE) DIGICERT(NINE)
                       NEWDIGIC(TEN)
                       FORCE
```

More information:

REKEY Function—Create Certificate from Existing Certificate (see page 84)
FORCE Keyword—Bypass Checks (see page 215)

UNLOCK Function—Unlock Online Terminal

Valid on z/OS, z/VSE, and z/VM.

Use the UNLOCK command function to unlock a terminal.

The UNLOCK command function:

- Is disabled if the NOLCFCHK attribute is attached to the ACID
- Cannot be routed through the security network with the CPF
- Prompts for the current sign-on password in z/VM, TSO, CICS, CA-IDMS, and CA-Roscoe

In TSO, the TSS UNLOCK command takes affect when going from one program to another. If you are in ISPF/EDIT, you can still ACCESS and ALLOCATE data sets to which you have access. The TSS LOCK/UNLOCK commands take affect when you leave EDIT/BROWSE.

An attempt to use a locked terminal:

- In z/VM, IMS, CA-IDMS/UCF, and CA-Roscoe, results in the immediate termination of the command or transaction. The only functions that can be performed from a locked terminal are unlock, signoff, WHOAMI, and other limited functions, such as TSO TIME, that do not access resources.
- In CICS and CA-IDMS/DC, the ACID is prompted for his password. If the password is valid the request is executed, if not the request is terminated.

This command function has the following format:

```
TSS UNLOCK
```

For IMS only, this command function has the following format:

```
TSS(password) UNLOCK
```

WHOAMI Function—Display ACID's Environment

Valid on z/OS, z/VSE, and z/VM.

Use the WHOAMI command function to display the:

- ACID
- ACID TYPE (USER, DCA, VCA, ZCA, LSCA, SCA)
- Security Mode (DORMANT, WARN, IMPL, FAIL)
- Facility
- Terminal Name
- Lock Time
- Logging Options
- Installation Data (z/OS and VSE only)

- Security Bypass Attributes (NOVOLCHK, NODSNCHK, etc.)
- Current identification of CPU
- If the ACID is undefined
- If the virtual machine is currently running under a surrogate ACID. (z/VM only)

The TSS WHOAMI command function:

- Will not display the terminal name when issued from a CRTE terminal
- Cannot be routed through the security network using CPF

This command function has the following format:

```
TSS WHOAMI
```

WHOHAS Function—Display ACID's Resource Access

Valid on z/OS, z/VSE, and z/VM.

Use the WHOHAS command function to display:

- All ACIDs, within the administrator's scope, that have access (using the ADDTO or PERMIT function) to a specified resource.
- All ACIDs, within the administrator's scope, that have access (using the ADDTO function) to a specified facility, field, or attribute.

Only one resource keyword and prefix may be specified per command. Attributes and non-resource fields, however, may be specified on a command. If several attributes, fields, or facilities are specified, all attributes, fields, or facilities must be simultaneously present.

The administrators must have:

- RESOURCE(INFO) authority, via the ADMIN function, to obtain access information for resources or job ACIDs owned within their scope
- FACILITY(facname) or FACILITY(ALL) authority, via the ADMIN function, to obtain access information for FACILITY for ACIDs within their scope

- MISC9(MODE) authority, via the ADMIN function, to obtain access information for MODES for ACIDs within their scope
- ACID(INFO) authority, via the ADMIN function, to obtain access information for job submission ACIDs to another ACID with their scope

Note: If mixed case is used for an FDT field or the HFSSEC resource class, when issuing a TSS WHOHAS for the FDT field or HFSSEC resource class, the case on the WHOHAS command must match the case in the FDT data or HFSSEC resource name on the acid in order for that data to be displayed.

This command function has the following format:

```
TSS WHOHAS keyword(value|*) [DATA(MASK|NOPREFIX|LITERAL)]
```

For queries on:

- A resource class, use the RDT RESCLASS
- A field in the FDT, use the FDTNAME.
- An CA Top Secret attribute, use the attribute name
- An CA Top Secret facility, use the keyword FACILITY

Enter the specific resource prefix, field value, attribute value (if applicable) or facility name for which access information is required or enter the data that must be contained in that field.

Optionally, limit the type of matching used for resource queries (LITERAL, MASK, NOPREFIX).

To display all permitted access to a maskable resource type, enter an asterisk (*) with DATA(MASK).

WHOHAS Applicable Keywords

This command function uses:

- All resources defined in the RDT. Additional resource keywords may be used by an installation that has defined new resource classes in the RDT record.
- Attributes and keyword(value) applicable keywords.
- Any field name defined to the FDT, both user-defined and predefined.
- The attributes that appear in the following lists.

List the FDT, to obtain FDT field names.

Use the following attributes, which take no value, with the TSS WHOHAS command:

ASUSPEND	MRO	NOOMVSDF	NOVOLCHK

AUDIT	MULTIPW	NOPWCHG	RSTDACC
CONSOLE	NOADSP	NORESCHK	SUSPEND
DUFUPD	NOATS	NOSUBCHK	TRACE
DUFXTR	NODSNCHK	NOSUSPEND	TSOMPW
GAP	NOLCFCHK	NOVMDCHK	

You must specify the following attributes with a value with the TSS WHOHAS command:

FACILITY (If specified with another field/attribute)	SMSAPPL	TSOLACCT	WAADDR1
	SMSDATA	TSOLPROC	WAADDR2
LANGUAGE	SMSMGMT	TSOLSIZE	WAADDR3
LTIME	SMSSTOR	TSOMCLASS	WAADDR4
OPCLASS	SNAME	TSOMSIZE	WABLDG
OPIDENT	SOURCE	TSOSCLASS	WADEPT
OPPRTY	TSOCOMMAND	TSOUDATA	WANAME
PHYSKEY	TSODEFPRFG	TSOUNIT	WAROOM
SCTYKEY	TSODEST	TZONE	
SITRAN	TSOHCLASS	UNAME	
	TSOJCLASS	WAACCNT	

Resource Access Information

Resource access information can be obtained by specifying a prefix, fully qualified name (within quotes), or a pattern containing masking characters. Not all resources support masking characters.

The amount of information displayed by the WHOHAS function can be voluminous depending upon the number of PERMITs defined. The DATA(option) keyword can be used to limit the display.

If you issue the WHOHAS command for DSNAME(SYS), it will return the OWNER for SYS1, then all of the authorizations under the owner. Next, you will get the owner for SYS2 and all of those authorizations until the list is complete.

Facility, Attribute, and Data Field Access Information

Facility access information can be obtained by specifying a fully qualified facility name; no prefix or masking is supported. Attribute information can be obtained by entering one or more attribute names. Data field information can be obtained by specifying the full data name.

Because facility information is not maintained as a resource, the amount of work required to obtain this information is dependent on the scope of the administrator requesting it. For a ZCA or lower, it is reasonably quick; however, for an SCA or an LSCA, it can require much longer to complete. Consider using batch processing to execute this command as an SCA. The amount of time required is greater than the time required to execute the TSS LIST(ACIDS) DATA(BASIC) command.

Examples: the WHOHAS function

This example lists all ACIDs that have the facility CICSPROD:

```
TSS WHOHAS FACILITY(CICSPROD)
```

This example lists all ACIDs that have the NODSNCHK attribute and simultaneously have the TSO procedure PROC999 as their default logon procedure:

```
TSS WHOHAS NODSNCHK
           TSOLPROC(PROC999)
```

This example displays all permitted access to the resource type dataset:

```
TSS WHOHAS DSN(*) DATA(MASK)
```

Administrative Authority Information

To obtain administrative authority information use the AUTHADM keyword and specify any of the administrative authorities that are specified with the TSS ADMIN command.

Because facility, field and attribute information is not maintained in the SECFILE ACID Index, a sequential search within the administrator's scope completes the query. For a user with a small number of ACIDs in their scope, it is reasonably quick; however, for an SCA or an LSCA, it can require much longer to complete. Therefore, you might consider using batch processing to execute this command as an SCA. The amount of time required is similar to the time required to execute the TSS LIST(ACIDS) DATA(BASIC) command.

Example: WHOHAS function

To display permitted accesses to resources, the administrator enters a TSS WHOHAS command. This command displays the owner of the resource, ACIDs who are authorized access to the resource, and administrator ACIDs who are permitted administrative authority over the resource.

To determine who has access information for data sets prefixed with SFT.CICS., enter:

```
TSS WHOHAS DSNAME(SFT.CICS)
```

CA Top Secret displays the ACIDs and the access information shown below.

```
RESOURCE      =  SFT.CICS.                OWNER(SFTDEPT)
  XAUTH       =  SFT.CICS.             ACID(SFTUSR1)
    ACCESS    =  UPDATE
  XAUTH       =  SFT.CICS.LOAD         ACID(SFTUSR2)
    ACCESS    =  UPDATE,CONTROL
  XAUTH       =  SFT.                  ACID(SFTMNGR)
    ACCESS    =  READ
  ADMIN       =  SFT.                  ACID(SFTDCA)
    ACCESS    =  ALL
  XAUTH       =  SFT.*.TEST            ACID(SFTMNGR)
    ACCESS    =  UPDATE,CONTROL
```

The figure above shows the resource owner (SFTDEPT), all matching access PERMITS (XAUTH), as well as administrator ACIDs to which the resource was permitted with the ACTION(ADMIN) keyword.

WHOOWNS Function—Display Resource Owners

Valid on z/OS, z/VSE, and z/VM.

Use the WHOOWNS command function to:

- Display the ACID that owns a specific resource
- Display the ACIDs that own each resource of a specific type

Additional resource keywords may be used by an installation that has defined new resource classes in the RDT Record.

Only one resource keyword and prefix may be specified per command.

Administrators must have:

- RESOURCE(INFO) authority, via the ADMIN function, to identify owners of resources within their scope
- MISC9(GENERIC) authority, via the ADMIN function, to use the WHOOWNS keyword(*) command function
- MISC9(MODE) authority, via the ADMIN function, to obtain ownership information for MODES

Note: If mixed case is used for the HFSSEC resource class, when issuing a TSS WHOOWNS for the HFFSEC resource class, the case on the WHOOWNS command must match the case of the HFSSEC resource name on the owning acid in order for that data to be displayed.

When WHOOWNS functions are entered for a resource, all resource prefixes which generically match the requested resource prefix are displayed in alphabetical order.

This command has the format:

```
TSS WHOOWNS keyword(prefix|TSS mode|*)
```

Keyword

If information is required for resource ownership, enter the keyword for that resource. If information is required for a security mode, enter MODE.

Prefix

Specifies the specific resource prefix, or CA Top Secret mode for which ownership identification is required. Enter an asterisk (*) to list all owners for all resources of this type.

WHOOWNS Applicable Keywords

This command function uses the keywords:

ABSTRACT, APPCPORT, APPCSI, APPCTP, APPLICATION
AREA, CACMD, CIMS, CPCMD ,CPU, DATABASE, DB2COLL,
B2DBASE, DB2PKG, DB2PLAN, DB2STOGP, DB2SYS
DB2TABLE, DB2TABSP, DBD, DCSS, DCT, DEVICESDIAGNOSE
DSNAME, FCT, FIELD, IBMFAC, IBMGROUP, IUCV, JCT,
JESJOBS, JESSPOOL, MGMTCLAS, MODE, NODES, OPCMD,
OPERCMDS, OTRAN, PANEL, PPT, PROGRAM, PSB, PSFMPL,
SDSF, SMESSAGESPI, STORCLAS, SUBSCHEM,
SYSCONS, TARGET, TERMINAL, TSOACCT, TSOAUTH,
TSOPRFG, TSOPROC, TST, UR1/UR2, USRCLASS, VMCF,
VMDIAL, VMMACH, VMMDISK, VMNODE, VMRDR, VOLUME,
VTAMAPPLDB2, VXDEVICEDB2BUFFP, VXFILE, WAIT, and WRITER.

Example: WHOOWNS function

This example displays who owns the SFT.CICS. data set:

```
TSS WHOOWNS DSNAME(SFT.CICS)
```

This example lists the owners of all of the division's data sets:

```
TSS WHOOWNS DSNAME(*)
```

Chapter 3: Keywords

ACCESS Keyword—Control Access to Resources

Valid on z/OS, z/VM, and z/VSE.

Use the ACCESS keyword to control the types and level of access to many resource classes such as:

- Data sets
- Volumes
- CICS, CA IDMS, and IMS resources
- User-defined resources (UR1 and UR2)

Ownership automatically confers total access to the resource to the owner unless Record Level Protection (RLP) is implemented.

This keyword has the following format:

`TSS PERMIT(`*`acid`*`) PPT(`*`pptname`*`) ACCESS(`*`access level`*`)`

This keyword can be used with:

- The commands ADMIN, PERMIT, and REVOKE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA

You can also use this keyword when adding a resource to the AUDIT record using the ADD command.

Access Levels

When granting XAUTH authority to the named “ACID,” the administrator may limit the access levels which the named “ACID” can PERMIT.

DEFAULT

If the ADMIN command does not specify an ACCESS clause, the named "ACID" of the command is not allowed to specify an ACCESS keyword in PERMIT commands. As a result, all PERMIT commands issued by the named "ACID" will default to the DEFACC access-level defined in the RDT

ALL

Named "ACID" may permit any resource at any access level.

CONTROL

Named "ACID" may permit any resource at the access level CONTROL.

CREATE

Named "ACID" may permit any resource at the access level CREATE.

DELETE

Named "ACID" may permit any resource at the access level DELETE.

FEOV

Named "ACID" may permit any resource at the access level FEOV.

FETCH

Named "ACID" may permit any resource at the access level FETCH.

NONE

Named "ACID" may permit any resource at the access level NONE.

PURGE

Named "ACID" may permit any resource at the access level PURGE.

READ

Named "ACID" may permit any resource at the access level READ.

REPLACE

Named "ACID" may permit any resource at the access level REPLACE.

SCRATCH

Named "ACID" may permit any resource at the access level SCRATCH.

UPDATE

Named "ACID" may permit any resource at the access level UPDATE.

WRITE

Named "ACID" may permit any resource at the access level WRITE.

Note: The ACLST of the resource class definition in the RDT governs the appropriate use of ACCESS levels in PERMIT commands for individual resources.

More information:

ACLST Keyword—Resource Access Level (see page 114)

Examples: ACCESS keyword

This example authorizes USER01 to update all data sets with AP as their highest-level qualifier:

```
TSS PERMIT(USER01) DSNAME(AP.) ACCESS(UPDATE)
```

In this example, USER02 must load the PAYUPDAT program from the PAYROLL.PRODLIB library, through BATCH to access the PAYROLL.MASTER data set:

```
TSS PERMIT(USER02) DSNAME(PAYROLL.MASTER) PRIVPGM(PAYUPDAT)
LIBRARY('PAYROLL.PRODLIB') FACILITY(BATCH) ACCESS(UPDATE)
```

ACID Keyword—Authorize ACIDs

Valid on z/OS, z/VSE, and z/VM.

Use the ACID keyword to:

- Give administrators the authority to specify the authority levels at which they can manage ACIDs within their scope
- Cross authorize an ACID to submit jobs under another ACID
- Use the ACID keyword on a MODEL command to create a new user ID modeled after the user ID specified by the USING keyword. The TSS commands produced by the MODEL command will contain the user ID specified by the ACID keyword.

This keyword has the following format:

```
TSS ADMIN(acid) ACID(authority level(s))
TSS PERMIT(acid) ACID(acid)
TSS MODEL ACID(acid) USING(newacid)  INTO(datasetname(membername))
```

This keyword can be used with:

- The commands ADDTO (STC only), ADMIN, DEADMIN, LIST, MODEL, PERMIT, REMOVE, RENAME, REVOKE, and WHOHAS
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ALL
- ACID(MAINTAIN) authority and DATA(ALL)

ACID Authority Levels

The CA Top Secret administrator can specify the authority levels:

ALL

Grants all of the ACID type authorities.

AUDIT

Allows an administrator to attach the AUDIT attribute to ACIDs within his scope. This authority level is required to report on the AUDIT attribute. An administrator without ACID(AUDIT) may not run TSSAUDIT and is not able to see the AUDIT attribute through TSS LIST.

CREATE

Gives an administrator the authority to CREATE and DELETE ACIDs.

DEFNODES

The ability to ADD, REMOVE, and REPLACE DEFNODES on an ACID record.

INFO

Allows administrators to use the WHOHAS function to determine who has access to specific ACIDs used for job submissions.

REPORT

Grants the administrator the authority to use the TSSUTIL and TSSCHART utilities to prepare customized reports showing the security activity for ACIDs within his scope.

MAINTAIN

Allows administrators to perform TSS command functions for ACIDs within their scope, including:

- RENAME—Renames an ACID.
- MOVE—Moves an ACID from one department or division to another. MOVE will work only if both the source organization and the destination are within the administrator's scope. MOVE authority is *not* valid if assigned to users or DCAs.
- REPLACE—Does one of the following:
  - Replaces the NAME of an ACID
  - Replaces the ACID's password information
  - Replaces the expiration date for an ACID via the FOR keyword
  - Replaces the CICS OPIDENT, OPPRTY, OPCLASS, PHYSKEY, and/or SITRAN values for an ACID
- ADDTO—Does one of the following:
  - Adds the SOURCE from which an ACID may initiate
  - Adds CICS OIDCARD and PHYSKEY values
  - Adds the profiles to which an ACID is connected
  - Adds the TSOMPW attribute that supports multiple passwords, on a user-by-user basis, in UADS

XAUTH

Gives the authority to PERMIT or REVOKE ACIDs used for job submissions.

Examples: ACID keyword

This example gives the Payroll Department's auditor the authority to generate TSSUTIL reports regarding ACID activity, and the authority to see who has access to the department's job submission ACIDs, via the WHOHAS function:

```
TSS ADMIN(PAYAUD) ACID(REPORT,INFO)
```

This example gives the Payroll Department's auditor the authority to prepare customized reports showing security activity:

```
TSS ADMIN(PAYAUD) RESOURCE(REPORT)
```

This example gives a DCA the authority to create, delete, and maintain the department's ACIDs, and to permit users in the department to use job submission ACIDs to submit jobs:

```
TSS ADMIN(TECHDCA) ACID(CREATE,MAINTAIN,XAUTH)
```

This example removes TECHDCA's authority for ACIDs:

```
TSS DEADMIN(TECHDCA) ACID(CREATE,MAINTAIN,XAUTH)
```

This example allows user WYLMI01 to submit a job under SMITH01:

```
TSS PERMIT(WYLMI01) ACID(SMITH01)
```

ACIDPRFX Keyword—List ACIDs With Prefix

Valid on z/OS, z/VM, and z/VSE.

Use the ACIDPRFX keyword to list all ACIDs that begin with a specified prefix. Prefixes can be from one to seven characters long.

This keyword has the following format:

```
TSS LIST(ACIDS) ACIDPRFX(acid-prefix)
                DATA(datatype(s))
                [TYPE(acidtype)]
                [ZONE|DIVISION|DEPARTMENT(acid)]
```

This keyword is used with:

- The LIST command
- The ACID types User, Profile, DEPARTMENT, DIVISION, ZONE, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ACIDS
- Data concerning ACIDs within the administrator's scope

Examples: ACIDPRFX keyword

This example lists all the ACIDs that begin with TEST:

```
TSS LIST(ACIDS) ACIDPRFX(TEST)
```

This example lists all data concerning all DCAs in ZONE(ACCOUNTG) whose ACID begins with ship:

```
TSS LIST(ACIDS) ACIDPRFX(ship)
                DATA(ALL)
                TYPE(DCA)
                ZONE(ACCOUNTG)
```

ACLST Keyword—Resource Access Level

Valid on z/OS, z/VSE, and z/VM.

Use the ACLST keyword to define, change, or remove access levels for the resource in the RDT Record.

This keyword has the following format:

```
TSS ADDTO(RDT) RESCLASS(resource Type)
               RESCODE(hex code)
               [ACLST(access level list)]
```

The access level list can consist of any combination of any user or CA Top Secret defined access levels. The access level mask is not required when specifying a standard access level name defined by CA Top Secret. However, the mask must be specified in order to redefine the hex-value of an CA Top Secret defined access level name to a non-standard access-level mask, or to define your own unique access level.

Common CA Top Secret defined access levels and their hexadecimal value are:

```
ALL=FFFF              MWRITE=2400
AUTOLOG=4000          MULTI=0400
BLP=8000              NOCREATE=0100
BROWSE=0200           NONE=0000
COLLECT=0002          NONSHR=2000
CONTROL=0400          PURGE=0100
CREATE=1000           READ=4000
DELETE=1000           REPL=0800
FEOV=0200             SCRTCH=0800
FETCH=8000            SHR=4000
FIND=1000             SUROGATE=2000
GRPLOGON=1000         UPDATE=8000
LOGON=8000            WRITE=2000
MREAD=4400
```

Notes:

- Access levels within the ACLST should be unique and in descending hexadecimal order so that permits with multiple access levels display the highest possible access level. For example, if the list specifies READ,UPDATE=6000; the UPDATE implies READ and a permit is issued for UPDATE access. The TSS LIST command will show that the permit includes READ access and will display READ instead of UPDATE. The access level is displayed correctly if the ACLST is specified as UPDATE=6000,READ.
- When adding a new resource class that contains a two byte RESCODE, or a one byte RESCODE with the ATTR(MASK) value set, a default ACLST is established if there is no ACLST keyword included in the definition. This ACLST will include two access levels, ALL(FFFF) and NONE(0000). Additionally, the default access level DEFACC is set to ALL(FFFF). These values can be modified with a TSS REPLACE command.
- Assigning an access level of ALL=FFFF protects the ALL access level as well as all other access levels that are defined in the ACLST. Security calls issued for undefined access levels will yield unpredictable results.
- All hexadecimal values must be unique. You cannot specify two different access levels with the same hexadecimal value.

- The hexadecimal values should be structured so that each binary position in the ACLST represents an independent attribute. Binary positions should only be re-used in the ACLST when one access-level is intended to include another. For instance, with the DATASET resource shown below, note that UPDATE uses the same bits in the hex-mask as were used by READ and WRITE, but that all other access levels in the list represent independent binary positions in the ACLST mask.
  ```
  ACLST(ALL,FETCH=8000,UPDATE=6000,READ=4000,WRITE=2000,CREATE=1000,
  SCRATCH=0800,CONTROL=0400,INQUIRE=0080,SET=0040,NONE)
  ```

  Granting permission with UPDATE access automatically confers READ and WRITE because the corresponding hex-values are additive, as shown next:
  ```
  UPDATE = READ + WRITE
   6000 = 2000 + 4000
  ```

  If you construct an access level CONFIRM which includes CREATE, SCRATCH, and CONTROL access, add the following to the current access list, between WRITE and CREATE, to preserve the descending order.
  ```
  CONFIRM=1C00=1000+800+400
  ```

  When using ACLST with a TSS REPLACE command, the entire access list is replaced by the values specified in the command.

This keyword is used with:

- The commands ADDTO and REPLACE
- The RDT ACID only
- MISC1(RDT) authority

Examples: ACLST keyword

This example adds a new resource called #PRODUCT to the RDT Record with READ and WRITE access levels:

```
TSS ADDTO(RDT) RESCLASS(#PRODUCT)
               ACLST(READ,WRITE)
```

This example uses the REPLACE function to add new access levels to a resource class already defined in the RDT Record:

```
TSS REPLACE(RDT) RESCLASS(#PRODUCT)
                 ACLST(READ,WRITE)
```

For unique access levels, which are more applicable to the resource, specify the hexadecimal values associated with each access level:

```
ACLST(ANSWER=0400,CALL=0200)
```

This example combines predefined values with the unique access levels:

```
ACLST(READ,ANSWER=0400)
```

This example uses the REPLACE command function to remove an access level list:

```
TSS REPLACE(RDT) RESCLASS(#PRODUCT)
                 ATTR(NOACCESS)
```

ACTION Keyword—Assign Actions

Valid on z/OS, z/VSE, and z/VM.

Use the ACTION keyword to assign actions:

- To an ACID when access to a FACILITY is attempted
- Taken when access to a resource is attempted

When used with FACILITY, this keyword has the following format:

```
TSS ADDTO(acid) FACILITY(facility)
                ACTION(AUDIT,NOTIFY,DENY)
```

AUDIT

CA Top Secret audits the ACID when logged on to the facility.

NOTIFY

CA Top Secret notifies the security console that the ACID is signing onto the facility.

DENY

CA Top Secret denies access to a facility even though it was specified in the ACID's PROFILE.

When used with resource, this keyword has the following format:

```
TSS PERMIT(acid) resource(prefix)
                 ACTION(FAIL,AUDIT,NOTIFY,DENY,VMPRIV,PASSWORD
                        ,NODSNCHK,ADMIN, EXIT, REVERIFY)

TSS REVOKE(acid) resource(prefix)
                 ACTION(ADMIN)
```

FAIL

If this permission is used by CA Top Secret as the "best fit" for the resource, CA Top Secret processes the request as if the user were in FAIL mode. CA Top Secret fails any unauthorized access to the resource, and conversely, allow authorized access (superseding, for example, native password protection on data sets and minidisks).

Note: FAIL can be used with resources that have access levels, whereas DENY is used with resources that do not have access levels.

AUDIT

CA Top Secret audits the access.

NOTIFY

CA Top Secret notifies the security console of resource access via the message TSS7299I.

EXIT

If the PERMIT is granted, CA Top Secret issues the EXIT call to invoke the TSS Installation Exit for data sets and volumes only in z/OS and for all z/VM resources.

DENY

Valid for all resources which do not support access levels, ACTION(DENY) fails any attempted access to the resource defined in the PERMIT. This ACTION effectively provides ACCESS(NONE) to the resource. The ACID's mode is still honored.

VMPRIV

(VMPRIVILEGE) The privileged form of CP commands and DIAGNOSE instructions.

PASSWORD

For minidisks and data sets only.

If the PERMIT is granted, CA Top Secret returns control to z/VM for password protection. This assumes that a link password exists for the minidisk.

Note: Any data set checks which occur as a result of the allocation of an SMS-managed data set is not prompted for a data set password. This is a normal function of SMS.

NODSN

For volumes only. Only volume checking is performed. All data set restriction is bypassed, and the minimum and maximum access levels to all data sets are controlled by the volume access with PERMIT.

ADMIN

Under normal circumstances, CA Top Secret allows an administrator to PERMIT and REVOKE only those resources which fall under his scope. The ACTION(ADMIN) keyword gives the security administrator the ability to allow ACIDs within his scope the authority to administer resources that are not within the permitted ACID's scope. If an access level is not specified, CA Top Secret permits the default access level for that resource class.

Note: ACTION(ADMIN) is not valid for Profile type ACIDs.

REVERIFY

Transactions that require additional security can be defined to require the signon password entered with each use. Use this to prevent sensitive transactions being entered by an unauthorized individual at an unlocked terminal.

This keyword is used with:

- The commands PERMIT, REVOKE (ACTION(ADMIN) only for REVOKE), REPLACE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ALL
- Authorization to administer a specific facility, via the ADMIN function, to authorize access to ACIDs
- Resource(XAUTH) authority to specify ACTION for resources that are owned within their scope

Examples: ACTION keyword

This example audits an ACID accessing CICSTEST:

```
TSS ADDTO(USER01) FACILITY(CICSTEST)
                  ACTION(AUDIT)
```

This example denies a user's attempts to logon to a facility:

```
TSS ADDTO(USER01) FACILITY(CICSPROD)
                  ACTION(DENY)
```

This example indicates that the user is allowed to sign on to CICS and everything the ACID does is audited:

```
TSS ADDTO(acid) FACILITY(CICS)
                ACTION(AUDIT)
```

In this example, USER01 is in WARN mode, and is not connected to PROF01, which allows access to MASTER.PAYROLL.FILE. Ordinarily, if USER01 attempted to access the MASTER.PAYROLL.FILE, he would be warned. This example fails USER01 if he attempts to access the MASTER.PAYROLL.FILE:

```
TSS PERMIT(USER01) DSNAME('MASTER.PAYROLL.FILE')
                   ACTION(FAIL)
                   ACCESS(NONE)
```

This example creates an audit record every time USER01 updates the PERS.PAY data set:

```
TSS PERMIT(USER01) DSNAME(PERS.PAY)
                   ACTION(AUDIT)
                   ACCESS(UPDATE)
```

This example issues message TSS7299I whenever USER01 accesses PERS.PAYROLL:

```
TSS PERMIT(USER01) DSNAME('PERS.PAYROLL')
                   ACCESS(R)
                   ACTION(NOTIFY)
```

In this example, USER01 is connected to a profile (PROF01) that allows him to access all terminals in the Accounting Department:

```
TSS PERMIT(PROF01)TERMINAL(K06L1234)
```

This example denies USER01 access to terminals K06L4567 and K06L1233:

```
TSS PERMIT(USER01)TERMINAL(K06L4567,K06L1233)
                  ACTION(DENY)
```

This example denies access and result in a failure in any mode:

```
TSS PERMIT(USER01)TERMINAL(K06L4567,K06L1233)
                  ACTION(FAIL,DENY)
```

This example allows USER05 to permit or revoke access to this resource to other users-although the resource itself is not owned within his scope.

```
TSS PERMIT(USER05) DSNAME(SYS1.)
                   ACTION(ADMIN)
```

This example removes authority from USER05 to permit or revoke access to this resource. The resource itself is not owned within his scope.

```
TSS REVOKE(USER05) DSNAME(SYS1.)
                   ACTION(ADMIN)
```

Note: If the FACILITY keyword had been specified on the above command, the command would have been processed without an error. However, FACILITY restrictions are not in effect.

In this example with OTRAN, password re-verification is indicated on the PERMIT by the ACTION(REVERIFY) parameter once the resource is owned:

```
TSS PERMIT(USER01) OTRAN(PROD)
                   ACTION(REVERIFY)
```

AFTER and BEFORE Keywords—Specify Profile Order

Valid on z/OS, z/VSE, and z/VM.

Use the AFTER and BEFORE keywords to add a profile to a specific location in the order of profiles.

This keyword has the following format:

```
TSS ADDTO(acid) PROFILE(new profile)
                AFTER|BEFORE(existing profile)
```

This keyword is used with:

- The ADDTO command
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Example: AFTER keyword

This example assumes profiles PROF01 through PROF04 already exist for ACID USER01, and PROF05 is added between PROF02 and PROF03:

```
TSS ADDTO(USER01) PROFILE(PROF05)
                  AFTER(PROF02)
```

The order of profiles for ACID USER01 is: PROF01, PROF02, PROF05, PROF03, PROF04.

ALTNAME Keyword—SubjectAltname Values

Valid on z/OS.

Use the ALTNAME keyword to specify the appropriate values for the SubjectAltname extension, of which one or more values might be coded. There is no default for ALTNAME.

This keyword has the following format:

```
TSS GENCERT(acid) ALTNAME('IP=nnn.nnn.nnn.nnn
                          DOMAIN=domain-name
                          EMAIL=email-address
                          URI=uri')
```

IP

Specifies a string containing a fully qualified numeric IP address. The format can be:

IPv4 dotted decimal

Four decimal numbers between 0 and 255 separated by periods. For example:

```
141.202.1.255
```

IPv6

Eight parts divided by colons with each part a hexadecimal number between 0 and FFFF. For example:

```
1080:23B4:324:4:3BCD:26:39F4:332
```

IPv4 compatible IPv6 address

A combination of the two, six parts of the IPv6 followed by the IPv4 address. For example:

```
0:0:0:0:0:FFFF:141.202.1.255
```

The maximum field size is 45 bytes.

DOMAIN

Specifies a string containing a fully qualified internet domain name. For example:
`ALTNAME(DOMAIN=CA.COM)`

EMAIL

Specifies a string containing a fully qualified email address. For example:
`ALTNAME('EMAIL=david@kindgom.net')`

URI

Specifies the universal resource identifier. For example:
`ALTNAME('URI=www.ca.com')`

Note: When you specify multiple parameters to ALTNAME, include one single quote at the beginning and end of the parameter list. Multiple parameters are separated with a space. For example:

```
ALTNAME('IP=200.100.10.1 EMAIL=my.email@test.net')
```

This keyword is used with:

- The GENCERT and RENEW commands
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority for users within their scope, and MISC4(CERTGEN).
- MISC4(CERTSITE) authority for CERTSITE ACID and MISC4(CERTAUTH) for CERTAUTH ACID.

Example: ALTNAME keyword

This example specifies the values EMAIL=DAVID@KINDGOM.and NET IP=123.123.123.123 values for the SubjectAltname extension:

```
TSS GENCERT(acid) DIGICERT(cert01)
                  SUBJECTN('CN=CERT01')
                  KEYUSAGE(HANDSHAKE)
                  ALTNAME('EMAIL=DAVID@KINDGOM.NET IP=123.123.123.123')
```

Note: Include single quotes if specifying more than one value with KEYUSAGE. For example:
`KEYUSAGE('HANDSHAKE DATAENCRYPT')`

APPLDATA Keyword—Associate Data with PERMIT

Use the APPLDATA to associate data with a particular PERMIT.

This keyword has the following format:

```
TSS PERMIT(acid) RESOURCE(prefix)
                 APPLDATA(data)
```

Example: APPLDATA keyword

This example associates the data ROLEMGR with a TSTUSER's permit for resource MANAGER in the EJBROLE resource class:

```
TSS PERMIT(TSTUSER) EJBROLE(MANAGER)
                    APPLDATA(ROLEMGR)
```

ARCHIVE Keyword—Archive a User's Security Record

Valid on z/OS.

Use the ARCHIVE keyword on a LIST or DELETE command to archive a user's permissions and resources into the form of TSS commands. The produced TSS commands can be stored in a PDS data set and used to restore the user in the future.

Important! Use ACID(MAINTAIN) and DATA(ALL) authority to archive the entire ACID. Anything less than DATA(ALL) provides only partial ACID archival. For example, DATA(BASIC) archive only the basic information.

This keyword has the following format:

```
TSS LIST(acid) ARCHIVE INTO(dataset_name[(member_name)])
TSS DELETE(acid) ARCHIVE INTO(dataset_name[(member_name)])
```

acid

Identifies the user ACID.

dataset_name

Specifies the name of a data set in which you want to store the security record information.

Important! You must specify a data set name that does *not* already exist. Additionally, you must have authorization to access the data set as specified in the command. For example, if you specify to archive into a USER01.SAMPLE data set, you must be authorized to data sets that start with USER01.

member_name

Specifies the name of a member in the specified data set.

Note: If *member_name* is omitted from the INTO keyword, the *acid* name is used as the member name. However, if the first character of the ACID name is a number, you must supply a member name that does not start with a number.

When you use the ARCHIVE keyword, the first record in the output is a comment containing the following information:

- Command
- User ACID being modeled
- Date and time of the archive
- TSS administrator who issued the command
- System on which the command was executed

The following example shows a comment:

```
/*ARCHIVE  CASSIE   STORED 02/17/10-09.54.00 BY MASTER1  ON XE15
```

This keyword is used with:

- The commands LIST and DELETE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority and DATA(ALL) authority

Example: Archive a User's Security Record

This example archives user security record information into a member named MYARCHIVE in the data set USER01.SAMPLE.ARCHIVE:

```
TSS LIST(CLRK99) ARCHIVE INTO(USER01.SAMPLE.ARCHIVE(MYARCHIVE))
```

Fields Not Copied during User Archive

During the archive process most of the user's security record information is archived. The following fields are *not* copied during the archive process:

- Field values not displayed by a TSS LIST command
- Passwords or passphrases
- Digital certificate and Keyring segments:
  - Certificate name, certificate start date, certificate until date, certificate ID, certificate subject DN
  - Certificate Keyring, certificate serial numumber, certificate issuer IDN, certificate issuer SDN, certificate NB date
  - Certificate NA date, certificate key size, certificate key type, certificate label, certificate trust status
  - Certificate URI, certificate IP address, certificate key usage
- Create/Modify date time
- Last used info
- ACID List entries
- ADMINBY
- Facility ADMINBY
- Facility Calendar
- Segment start

Note: If the user being archived has digital certificates, we highly recommend that the security administrator use the EXPORT command to export all certificate(s) and private key(s) in the PKCS12 format into a dataset. This exported certificate dataset can then be used to restore those certificates back to the user.

ASSIZE Keyword—Maximum Address Space Size

Valid on z/OS.

Use the ASSIZE keyword to specify the maximum address space region size allowed per process created via rlogin or telnet. This field overrides the MAXASSIZE parameter in the BPXPRMxx member of PARMLIB for this user.

ASSIZE is equivalent to ASSIZEMAX in RACF.

This keyword has the following format:

```
TSS ADD(acidname) ASSIZE(nnnnnnnn)
```

nnnnnnnn

The maximum address space region size allowed per process.

Range: 10,485,760 to 2,147,483,647

Default: USS takes the system defaults set in BPXPRMxx.

This keyword is used with the ADDTO, REPLACE, REMOVE, and WHOHAS command.

Examples: ASSIZE keyword

This example assigns an acid a value of 10,485,760 for ASSIZE:

```
TSS ADD(TESTID) ASSIZE(10485760)
```

This example removes ASSIZE from the acid:

```
TSS REMOVE(TESTID) ASSIZE
```

ASUSPEND Keyword—Remove Suspension

Valid on z/OS, z/VSE, and z/VM.

Use the ASUSPEND keyword to remove the suspension of an ACID that was suspended for administrative reasons.

This keyword has the following format:

```
TSS REMOVE(acid) ASUSPEND
```

This keyword is used with:

- The REMOVE command
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(REMASUSP) authority
- Administrators who do not have MISC8(REMASUSP) authority can remove an ASUSPEND for users within their scope if given UPDATE access to entity TSSCMD.USER.REMOVE.ASUSPEND in the CASECAUT resource class

Examples: ASUSPEND keyword

This example removes the ASUSPEND attribute from an ACID:

```
TSS REMOVE(ACARP) ASUSPEND
```

This example removes a temporary suspension that was added with a FOR or an UNTIL keyword, add the FOR or UNTIL keyword onto an ASUSPEND entry. The date or time period specified with a FOR or an UNTIL keyword on a temporary suspension, is not included on the matching ASUSPEND entry.

This example unsuspends USER01 if it was suspended with FOR:

```
TSS REMOVE(USER01) ASUSPEND FOR
```

This example unsuspends USER01 if it was suspended with UNTIL:

```
TSS REMOVE(USER01) ASUSPEND UNTIL
```

ASYMUSE Keyword—Asymmetric Key Usage Options

Valid on z/OS.

Use the ASYMUSE keyword to specify the appropriate values for the asymmetric key usage. If no value is specified, the key is eligible for all uses.

This keyword has the following format:

```
TSS PER(acid) CSFKEYS(resource) ASYMUSE(HANDSHAKE|SECXPORT)
```

HANDSHAKE|NOHANDSHAKE

Specifies whether the key can be used to protect communication channels. If not specified, HANDSHAKE is assumed.

SECXPORT|NOSECXPORT

Specifies whether the key can be used to export or import symmetric keys. If not specified, SECXPORT is assumed.

Note: Include single quotes if specifying more than one value with KEYUSAGE. For example:

```
ASYMUSE('HANDSHAKE SECXPORT')
```

This keyword is used with:

- The PERMIT command
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- CSFKEYS and XCSFKEY resource classes only
- Resource(XAUTH) authority to specify ACTION for resources that are owned within their scope

ATTR Keyword with the FDT—FDT Field Attributes

Valid on z/OS, z/VSE, and z/VM.

Use the ATTR keyword to add or replace attributes to a field in the FDT Record.

This keyword has the following format:

```
TSS ADDTO(FDT) FDTNAME(field name)
               FDTCODE(hex code)
               MAXLEN(40)
               ATTR(MIXED,NONDISP,NOXTRPRP,XTRPRP,attribute)
```

MIXED

Allows display of mixed-case data for a field in the FDT Record. If ATTR(MIXED) is not given, the information is added to the user in uppercase.

NONDISP

Prohibits display of a field when an administrator issues a TSS LIST command on the ACID. The data can be extracted or modified using the Application Interface or the RACROUTE macro, but it does not display using TSS LIST.

NOXTRPRP

Prohibits successful extract/replace call results for the field from being sent to the recovery file and propagated to other systems. You can change this attribute via a “TSS REPLACE” command for user-defined fields in the FDT and pre-defined fields in the FDT where CA did not already include this attribute.

XTRPRP

Allows successful extract/replace call results for the field to be sent to the recovery file and propagated to other systems. This attribute does not display when you list the field. You can change this attribute via a “TSS REPLACE” command for user-defined fields in the FDT and pre-defined fields in the FDT where CA did not already include this attribute or NOXTRPRP.

This keyword is used with:

- The commands ADDTO and REPLACE
- The RDT and FDT records
- MISC1(RDT) authority

The following reserved fields are permanently set to NOXTRPRP:

- EXTRPASS
- DATEPASS
- INTPASS
- ASISPASS
- TUPT
- TSOUDATA
- TSOOPT
- TRBA
- TCONS
- FLAG4
- ENCTYPE
- PHRASE
- LNXENTS
- PHRDATE
- PPNCY
- DCEKEY
- PWVCNT
- KERBPASS
- CURKEY
- CURKEYV
- PREVKEY
- PREVKEYV
- DCENCRY
- KEYSMSTR
- CURDES3
- PREDES3

- PREDES3V
- CURDESD
- CURDESDV
- PREDESD
- PREDESDV
- ADMPSWD
- EIMDOMAIN
- EIMOPTION
- EIMLOCREG
- PRXLDAPHST
- PRXBINDDN
- PRXBINDPW
- PRXBPWKEY
- PRXKRBREG
- PRX509REG
- CURAES128
- CURAES128V
- PREAES128
- PREAES128V
- CURAES256
- CURAES256V
- PREAES256
- PREAES256V
- WITHLABL
- ALTNAME

Example: ATTR Keyword

This example adds $TAX information to an ACID in mixed case:

```
TSS ADDTO(FDT) FDTNAME($TAX)
               FDTCODE(44)
               MAXLEN(40)
               ATTR(MIXED)
```

ATTR Keyword with the RDT—Resource Attributes

Valid on z/OS, z/VSE, and z/VM.

Use the ATTR keyword to define or change a resource to the RDT Record with one or more attributes.

This keyword has the following format:

```
TSS ADDTO(RDT) RESCLASS(resource Type)
               RESCODE(hex code)
               ATTR(attribute)
```

This keyword is used with:

- The commands ADDTO and REPLACE
- The RDT and FDT records
- MISC1(RDT) authority

Attributes

When used with the RDT, the following attributes can be used:

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

Changes a generic attribute to a non-generic one. A non-generic attribute only supports general resources (resources which do not support masking). It treats the specific resources within the resource class as fully qualified names. EXIT, DEFPROT, MERGE, ALLMERGE, and GENERIC are the only site modifiable attributes which can be changed for predefined resources.

PRIVPGM

Supports privileged program for this resource class.

NOPRIVPGM

Deactivates privileged program support.

LIBRARY

DATASET only. Restricts both the program allowed to access the data set, as well as the library from which the program is fetched.

NOLIB

PIE only. Deactivates support for a library with privileged program.

LONG

Equivalent to MAXLEN(44): indicates that the maximum length resource name for PERMIT with this RESCLASS is 44.

SHORT

(default) Equivalent to MAXLEN(8): indicates that the maximum length resource name for PERMIT with this RESCLASS is 8.

MERGE

Overrides the AUTH control option and uses the MERGE algorithm to determine the processing record for security validation of this RESCLASS.

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.

NOALLMERGE

Removes the ALLMERGE attribute from the resource.

VMUSER

Supports VMUSER for this resource class

NOVMUSER

Deactivates VMUSER support.

MASK

Allows the interpretation of MASK characters for this resource class in any facility set with the RES sub-option.

Altering a resource class to MASK from NOMASK makes existing resources un-administrable. Before making such a change, it is recommended that all existing permissions and ownerships be revoked and removed, then re-administered after the attribute change.

NOMASK

Prevents the interpretation of MASK characters in this resource class.

To pick up any permits on any acids in storage, including the ALL record, after altering NOMASK to MASK, refresh the acids. To refresh the ALL record, issue a dummy PERMIT command against the ALL record and then REVOKE the permit. To refresh a user acid in all address spaces where the user is signed on, enter:

```
TSS REFRESH(acid) JOBNAME(*).
```

SIGNAL

Specifies that CA Top Secret will issue an ENF62 signal when you add or remove ownership of this resclass to an ACID.

NOSIGNAL

```
Deactivates the SIGNAL for a resource class.
```

For PRIVPGM, LIBRARY, and VMUSER the security driver must also support these features. For a user-defined resource, the software that generates the security calls must supply additional parameters in order to satisfy PRIVPGM, LIBRARY, and VMUSER restrictions.

Notes:

- All of the operands used to deactivate an attribute can only be used with a TSS REPLACE(RDT) command function.
- The following default attributes are in effect when adding user—defined resource classes to the RDT using a RESCODE of 101—13F unless the corresponding overriding attribute is specified: MASK, NOEXIT, NODEFPROT, NOPRIVPGM, NOMERGE, NOALLMERGE, NOVMUSER, and GENERIC.
- The following default attributes are in effect when adding user—defined resource classes to the RDT using a RESCODE of 01—3F unless the corresponding overriding attribute is specified: NOMASK, NOEXIT, NODEFPROT, NOPRIVPGM, NOMERGE, ALLMERGE, NOVMUSER,GENERIC, and SHORT.

- 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 predefined resource classes, only EXIT, DEFPROT, MERGE, ALLMERGE, and GENERIC attributes can be changed.
- If mixed case is used for the HFSSEC resource class, when issuing a TSS WHOHAS for the HFSSEC resource, the case on the WHOHAS command must match the case in the HFSSEC resource on the acid in order for that data to be displayed. The same is true for TSS WHOOWNS.
- For the EXIT attribute to take effect the global control option EXIT(ON) must be set.
- For PRIVPGM, LIB, and VMUSER the security driver must also support these features. For a user-defined resource, the software that generates the security calls must supply additional parameters in order to satisfy PRIVPGM, LIB, and VMUSER restrictions.
- If mixed case is used for the FDT field, when issuing a TSS WHOHAS for the FDT field, the case on the WHOHAS command must match the case in the FDT data on the acid in order for that data displayed.

Examples: ATTR keyword

This example adds #PRODUCT to the RDT Record and gives it default protection:

```
TSS ADDTO(RDT) RESCLASS(#PRODUCT)
               RESCODE(002)
               ATTR(DEFPROT)
```

This example removes default protection from resource class #PRODUCT:

```
TSS REPLACE(RDT) RESCLASS(#PRODUCT)
                 ATTR(NODEFPROT)
```

This examples indicates SIGNAL=YES on the DATASET resclass, which tells the command processor to issue the enf62 signal to CSF when a DATASET is added or removed:

```
TSS REP(RDT) RESCLASS(DATASET) ATTR(SIGNAL)
```

To remove the LONG attribute which is already attached to a specific resource class, specify ATTR(SHORT).

AUDIT Keyword—ACID Activity

Valid on z/OS, z/VSE, and z/VM.

Use the AUDIT keyword to allow an audit of ACID activity.

Note: AUDIT is a reserved ACID name.

When used with a permission, this keyword has the following format:

```
TSS ADDTO(acid) AUDIT
```

Capacity of list

One to five resource names per CA Top Secret command.

When used as a reserved ACID, this keyword has the following format:

```
TSS ADDTO(AUDIT) resource class(resource name)
                 [ACCESS(level1, level2, …)]
```

Notes:

- Any masking character in a resource name added to the AUDIT Record is treated as a literal. For this reason resource names containing masks should not be added to the AUDIT Record.
- A resource defined to the AUDIT record cannot exceed 64 characters.
- ACCESS(NONE) is ignored. If access is not specified, ACCESS(ALL) is assumed.
- If the resource class in the RDT has the default GENERIC attribute, to audit only the non-generic resource name enclose the resource name in apostrophes with a trailing space.

This keyword is used with:

- The commands ADDTO, CREATE, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(AUDIT) authority to ADDTO or REMOVE the AUDIT attribute from ACIDs within their scope
- RESOURCE-AUDIT authority when using AUDIT as a reserved ACID name

Examples: AUDIT keyword

This example adds the audit attribute to an user01. Any actions performed by user01 are recorded in the Audit file for further review.

```
TSS ADD(user01) AUDIT
```

This example removes the AUDIT attribute from USER01:

```
TSS REMOVE(USER01) AUDIT
```

In this example, the program "LIST" is audited, but the program "LISTA" is not audited:

```
TSS ADD(AUDIT) PROGRAM('LIST ')
```

This example allows user02 to administer the audit attribute for ACIDs within his scope.

```
TSS ADMIN(user02) RESOURCE(AUDIT)
```

This example allows any activity for the specified resource class to be audited:

```
TSS ADD(AUDIT) resource class(resourcename)
```

BIND Keyword—Bind a Certificate to a PKCS#11 Token

Valid on z/OS.

Use the BIND keyword with the P11TOKEN function to bind a digital certificate to a z/OS PKCS#11 token. When a certificate is bound to a token, CA Top Secret creates a certificate, public key and private key (if the certificate has one and the BIND usage is personal).

The following key types are not supported:

- DSA public keys
- DSA, ICSF and PCICC private keys

The command has the following format:

```
TSS P11TOKEN BIND
             LABLCTKN(token name)
             TOKNDATA(userid,digicert)|LABLCERT(certificate label)|
             TOKNUSER(userid)
             [USAGE(PERSONAL|CERTSITE|CERTAUTH]
             [DEFAULT]
```

LABLCTKN(token name)

Specifies the name of the token. The token must already exist.

TOKNDATA(userid,digicert)

Userid specifies the ACID for the digital certificate. Digicert identifies the digital certificate. LABLCERT and TOKNDATA are mutually exclusive.

LABLCERT(certificate label)

Specifies the digital certificate label name of the certificate to bind. The userid for the certificate may be specified with the TOKENUSER keyword. If TOKNUSER is not specified then the userid of the administrator that issued the command is used. LABLCERT and TOKNDATA are mutually exclusive.

TOKNUSER(userid)

Specifies the userid for the digital certificate

USAGE[PERSONAL|CERTSITE|CERTAUTH]

Specifies how this certificate is used within the token. If usage is not specified, it defaults to the usage value of the certificate being bound.

DEFAULT

Specifies that the certificate is the default certificate for the token. Only one certificate may be the default. If a default certificate already exists, its default status is removed.

BIND Authority Levels

The administrator must have:

- MISC3(PTOK) authority
- MISC4(CERTSITE) authority when the certificate is from the CERTSITE acid
- MISC4(CERTAUTH) authority when the certificate is from the CERTAUTH acid

Controlled by ICSF using resources in the CRYPTOZ and IBMFAC resource classes.

To bind a certificate to a PKCS#11 token for USAGE personal use, the authority required is:

- For one's own certificate—Sufficient authority to CRYPTOZ resources and READ authority to IRR.DIGTCERT.BIND
- For someone else's certificate—Sufficient authority to CRYPTOZ resources and UPDATE authority to IRR.DIGTCERT.BIND
- For CERTSITE or CERTAUTH certificate—Sufficient authority to CRYPTOZ resources and CONTROL authority to IRR.DIGTCERT.BIND

To bind a certificate to a PKCS#11 token for USAGE CERTSITE or CERTAUTH use the authority required:

- For one's own certificate is sufficient authority to CRYPTOZ resources, CONTROL authority to IRR.DIGTCERT.ADD and READ authority to IRR.DIGTCERT.BIND
- For someone else's certificate is sufficient authority to CRYPTOZ resources, CONTROL authority to IRR.DIGTCERT.ADD and UPDATE authority to IRR.DIGTCERT.BIND
- For CERTSITE or CERTAUTH certificate is sufficient authority to CRYPTOZ resources and UPDATE authority

Example: BIND keyword

This example binds the certificate CERT001 to the token TOKEN#1:

```
TSS P11TOKEN BIND LABLCTKN(token#1) TOKENDATA(user01,cert0001)
```

BPECC Keyword—Generate Key with BPECC

Valid on z/OS

Use the BPECC keyword to generate a key pair using the brainpool ECC algorithm instead of the RSA algorithm. This parameter cannot be used with the ICSF, DSA, or NISTECC parameters. Because the key is generated using ICSF PKCS #11 functions, the ICSF subsystem must be operational and configured for PKCS #11 operation.

Minimally, ICSF must be at the HCR7770 level. The private key is placed in the CA Top Secret database. When ICSF is at the HCR7780 level or higher, specify PCICC or PKDSLBL to store the private key in ICSF instead of the CA Top Secret database.

This keyword has the following format:

```
TSS GENCERT(acid) DIGICERT(8-byte name) BPECC
```

The keyword is used with:

- The GENCERT and REKEY commands
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority for users, MISC4(CERTSITE) for CERTSITE ACID, and MISC4(CERTAUTH) for CERTAUTH ACID

CALENDAR Keyword—Create and Use a Calendar

Valid on z/OS, z/VSE, and z/VM.

Use the CALENDAR keyword to:

- Provide a name for CA Top Secret to reference a CALENDAR record
- Add, remove, replace, or list an existing calendar in the SDT Record
- Permit or revoke a CALENDAR to an ACID (When defined to the SDT)

When used with ADDTO, this keyword has the following format:

```
TSS ADDTO(SDT) CALENDAR(calname)
               [YEAR(yyyy)]
               [DAYS(daywk1,,,)]
               [INCLUDE(mm/dd)]
               [EXCLUDE(mm/dd)]
```

calname

Specifies a record ID that must be unique for each calendar. It can contain letters, numbers, and special characters.

Length: 8 characters

DAYS

Specifies the days to include in the calendar.

INCLUDE

Specifies the days to include in the calendar.

EXCLUDE

Specifies the days to exclude from the calendar.

When used with PERMIT, this keyword has the following format:

```
TSS PERMIT(acid) RESOURCE(resource-name)
                  ACCESS(level)
                 CALENDAR(calname
```

ACCESS

Specifies one of following access levels: ALL, CREATE, CONTROL, FETCH, NONE, READ, SCRATCH, UPDATE, or WRITE.

Default: READ

This keyword is used with:

- The commands ADDTO, DELETE, PERMIT, REMOVE, REPLACE, REVOKE, and LIST
- The SDT Record exclusively

- MISC3(SDT) authority to ADDTO or REMOVE records or fields in the SDT
- MISC8(LISTSDT) authority to list calendars in the SDT only
- Access control methods expiration, facility, and actions

Examples: CALENDAR keyword

This example creates a calendar for the present year called CAL1:

```
TSS ADDTO(SDT) CALENDAR(CAL1)
```

This example creates a calendar called CAL1 for the days Monday through Friday in the year 2009:

```
TSS ADDTO(SDT) CALENDAR(CAL1)
               YEAR(2009)
               DAYS(WEEKDAY)
```

This example permits a user to update the inventory master file data set INV.MASTER.FILE with the personnel department's CALENDAR CAL1:

```
TSS PERMIT(USR01) DSNAME('INV.MASTER.FILE')
                   ACCESS(UPDATE)
                  CALENDAR(CAL1)
```

This example revokes access:

```
TSS REVOKE(USR01) DSNAME('INV.MASTER.FILE')
```

CATEGORY Keyword—Category Specification

Valid on z/OS.

Use the CATEGORY keyword to define or remove categories to isolate users, data and resources within the organization. Categories are optional non-hierarchical elements of security labels.

This keyword has the following format:

```
TSS ADDTO(MLS) CATEGORY(category name)
```

Category name

Specifies the alphanumeric name of a category in the system. A category name can contain spaces.

Range: 1 to 32 characters

This keyword is used with:

- The commands ADDTO, LIST, and REMOVE
- MISC5(MLS) authority

Examples: CATEGORY keyword

This example adds category PAYROLL to the MLS Record:

```
TSS ADDTO(MLS) CATEGORY(ACCOUNTING)
```

This example removes the category ACCOUNTING:

```
TSS REMOVE(MLS) CATEGORY(ACCOUNTING)
```

CERTFLAG Keyword—Set or Reset Digital Certificate Flags

Valid on z/OS.

Use the CERTFLAG keyword to set or reset digital certificate flags.

This keyword has the following format:

```
TSS REPLACE(acid) DIGICERT(name)
                  CERTFLAG(NOGENREQ|GENREQ)
```

acid

Specifies the ACID affected by the REPLACE function.

name

Specifies a case-sensitive character ID that identifies the digital certificate to an ACID.

Length: 1 to 8 characters

NOGENREQ

Turns off the GENREQ indicator for a digital certificate that has had a GENREQ issued for it. Use this specification only in the following situations:

- You specified GENREQ when REKEY was intended.
- You specified REKEY when GENREQ was intended.

During normal processing, inserting a signed certificate over the GENREQ'd certificate turns off the GENREQ indicator.

GENREQ

Indicates that the CERTDATA record has been the target of a GENREQ command. We recommend *not* using the change command to turn on this indicator, because the GENREQ command will turn on the indicator.

This keyword is used with:

- The REPLACE command
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority for users
- MISC4(CERTSITE) authority for CERTSITE ACIDs
- MISC4(CERTAUTH) authority for CERTAUTH ACIDs

Example: Reset a Flag for a Digital Certificate

This example resets a flag for a digital certificate by turning off the GENREQ indicator:

```
TSS REPLACE(USER1) DIGICERT(TEST) CERTFLAG(NOGENREQ)
```

CERTMAP Keyword—Certificate Filter Association

Valid on z/OS.

Use the CERTMAP keyword to associate a certificate name filter with an ACID and define the filter. The TSS ADDTO CERTMAP command identifies the ACID assigned to a user if a user's digital certificate matches the subject's distinguished name filter and/or the issuer's distinguished name filter specified in the command.

When the MULTIID special ACID is used with the CRITERIA keyword, additional criteria select the ACID. The additional criteria is defined using the TSS ADDTO CRITMAP command.

CERTMAP specifies a unique eight-byte record identifier. When a certificate name filter is defined, the filter map information is stored in a CERTMAP record in the SDT on the security file.

This keyword has the following format:

```
TSS ADDTO(acid) CERTMAP(recid)
                SDNFILTR('subject-dist-name-filter')
                IDNFILTR('issuer-dist-name-filter')
                [LABLCMAP('32-byte label')]
                [DCDSN(data set name)]
                [TRUST|NOTRUST]
```

When criteria in addition to the distinguished name is used to assign an ACID to a user, this keyword has the following format:

```
TSS ADDTO(MULTIID) CERTMAP(recid)
                   SDNFILTR('subject-dist-name-filter')
                   IDNFILTR('issuer-dist-name-filter')
                   CRITERIA(criteria-name-template)
                   [LABLCMAP('32-byte label')]
                   [DCDSN(data set name)]
                   [TRUST|NOTRUST]
```

IDNFILTR

Specifies the significant portion of the issuer's distinguished name. In order for a match, when IDNFILTR is specified with DCDSN, the filter must correspond to a starting point within the issuer's distinguished name found in the certificate contained in the data set. The distinguished name from the point of the match to the end of the name is used as the filter data. If DCDSN is not specified, the entire portion of the issuer's distinguished name must be specified in full. The value must be enclosed in single quotes.

IDNFILTR is optional if SDNFILTR is specified. If IDNFILTR is not specified, only the subject's name is used as the filter. If IDNFILTR is specified and only a portion of the issuer's name is used as the filter, SDNFILTR must not be specified. If both IDNFILTR and SDNFILTR are specified, the IDNFILTR value does not need to begin with a valid prefix. This allows the use of certificates from a certificate authority that chooses to include non-standard data in the issuer's distinguished name.

A maximum of 1024 characters can be entered for IDNFILTR. When a starting value is specified for a certificate contained in a data set, there cannot be more than 1024 characters between the starting point and the end of the issuer's name in the certificate.

SDNFILTR

Specifies the significant portion of the subject's distinguished name. In order for a match, when SDNFILTR is specified with DCDSN, the filter must correspond to a starting point within the subject's distinguished name found in the certificate contained in the data set. The distinguished name from the point of the match to the end of the name is used as the filter data. If DCDSN is not specified, the entire portion of the subject's distinguished name must be specified in full. The value must be enclosed in single quotes.

SDNFILTR is optional if IDNFILTR is specified. If SDNFILTR is not specified, only the issuer's name is used as filter. SDNFILTR must not be specified with IDNFILTR unless the value of IDNFILTR will result in the entire issuer's name being used in the filter. The subject's name can be partial but cannot be used in a filter that contains only a partial issuer's name.

A maximum of 1024 characters can be entered for SDNFILTR. When a starting value is specified for a certificate contained in a data set, there cannot be more than 1024 characters between the starting point and the end of the issuer's name in the certificate.

DCDSN

Specifies the MVS data set containing the digital certificate. The data set must be defined as physical sequential (DSORG=PS) and variable blocked (RECFM=VB). The data set name is entered as a fully qualified name without enclosed quotes. The data set must be cataloged and up to 44 characters long.

The certificate contained in the data set must be BER-encoded, PKCS-7 BER-encoded, or Privacy Enhanced Mail (PEM)-encoded. PEM certificates must be transported to MVS as TEXT; the other formats must be transported as BINARY. The length of the serial number and certificate authority distinguished name must be less than 246.

This is a sample DCDSN command:

```
TSS ADD(USER01) DIGICERT(DIGI0001)
                DCDSN(USER01.CERTIF.001)
```

CRITERIA

Indicates that criteria in addition to the distinguished name is used to assign an ACID to a user. The CRITERIA keyword is used only with MULTIID. If MULTIID is not specified, any CRITERIA field is ignored.

LABLCMAP

(Optional) If it is not included in the command, the record identifier name entered for CERTMAP is automatically used for LABLCMAP.

TRUST|NOTRUST

A filter can be used to associate an acid with a certificate only when TRUST is specified.

Default: NOTRUST

This keyword is used with:

- The commands ADDTO, LIST, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Examples: CERTMAP keyword

In this example, users who enter the system with a certificate subject that starts with 'OU=NJ.OU=Sales.O=ABC Co' are assigned the accessor id NJDEPT1 if the certificate was issued by the VeriSign certificate authority.

If the subject matched, but the certificate was issued by another certificate authority, the user is assigned NJDFLT.

```
TSS ADDTO(NJDEPT1) CERTMAP(NJMAP1)
                   LABLCMAP('NJ Dept 1 Map')
                   TRUST
                   IDNFILTR('OU=VeriSign Class 1 Individual
                            Subscriber.O=VeriSign,
                            Inc.L=Internet')
                   SDNFILTR('OU=NJ.OU=Sales.O=ABC Co')

TSS ADDTO(NJDFLT) CERTMAP(NJDFLT)
                  LABLCMAP('NJ Dept 1 User')
                  TRUST
                  SDNFILTR('OU=NJ.OU=Sales.O=ABC Co')
```

In this example, users who enter the system with a certificate subject that starts with 'OU=Dept3.OU=NY.OU=Sales.O=ABC Co' are assigned to NJDEPT3:

```
TSS ADDTO(NYDEPT3) CERTMAP(NJMAP3)
                   LABLCMAP('NY Dept 3 Map')
                   TRUST
                   SDNFILTR('OU=Dept3.OU=NY.OU=Sales.O=ABC Co')
```

In this example, the application id criteria in addition to the distinguished name are used to determine which ACID to assign. Users in NY sales department Dept2 that handle corporate accounts using application BUSINESS to access the system, are assigned accessor id NYDEPT2B. Users that handle retail accounts using application RETAIL to access the system, is assigned to NYDEPT2R.

The special acid name of MULTIID along with the CRITERIA parameter tells CA Top Secret that if the subject and/or the issuer name information matches, then search the CRITMAP records for a match on application name before assigning an ACID to the user.

```
TSS ADDTO(MULTIID) CERTMAP(NYMAP2)
                   LABLCMAP('NY Dept 2 Map')
                   TRUST
                   SDNFILTR('OU=Dept2.OU=NY.OU=Sales.O=ABCCo')
                   CRITERIA(CNFAPP=&CNFAPP)

TSS ADDTO(NYDEPT2B) CRITMAP(NYCRIT2B)
                    CNFAPP(BUSINESS)

TSS ADDTO(NYDEPT2R) CRITMAP(NYCRIT2R)
                    10CNFAPP(RETAIL)
```

CERTNSER Keyword—Next Certificate Serial Number

Valid on z/OS.

Use the CERTNSER keyword to specify the next serial number used by this certificate to sign another certificate. Do not modify this number manually because you can generate duplicate certificates which are unusable on z/OS systems.

This keyword has the following format:

```
TSS REPLACE(acid) DIGICERT(8-byte name)
                  CERTNSER(nnnnnnnnnnnnnnnn)
```

nnnnnnnnnnnnnnnn

The hex value of the next serial number used by this certificate to sign another certificate. Every byte must be specified, including leading zeros.

Size: 16 bytes

This keyword is used with

- The REPLACE command only
- ACID(MAINTAIN) authority for ACIDs, MISC4(CERTSITE) for CERTSITE ACIDs, and MISC4(CERTAUTH) for CERTAUTH ACIDs
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA

Example: CERTNSER keyword

To update a certificate's next serial counter to 09, specify the full 16-byte serial number, which will be used as the starting value for the next serial number. The next time this certificate (AUTHCA01) is used in signing another certificate, the serial number generated will start again at 09.

```
TSS REPLACE(CERTAUTH) DIGICERT(AUTHCA01)
                      CERTNSER(0000000000000009)
```

CHAIN Keyword—Display Certificate Information

Valid on z/OS.

Use the CHAIN keyword to display information about a certificate owned by a user, CERTSITE, or CERTAUTH and its issuer’s certificates owned by user CERTAUTH in a chain of certificates.

This keyword is used with:

- The commands CHKCERT and LIST
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA

Example: Display Information about Digital Certificates in a Data Set

This example displays information about digital certificates in a data set:

```
TSS CHKCERT DCDSN(MYCERT.CERTDATA) CHAIN
```

Example: Display Information about Digital Certificates and Issuer's Certificates

This example displays information about digital certificates and issuer's certificates:

```
TSS LIST(USER1) DIGICERT(MYCERT1) CHAIN
```

CHKADDRS Keyword—Check Ticket Address

Valid on z/OS.

Use the CHKADDRS keyword to enables address checking in tickets for a Kerberos server running on z/OS 1.13 or higher. This field can be enabled only for the local realm (that is, when REALM(KERBDFLT) is specified).

This keyword has the following format:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
    REALMNAME('kerberos-realmname')
    MINTKTLF(min-ticket-life)
    MAXTKTLF(max-ticket-life)
    DEFTKTLF(nnnnnnnnnn)
    KERBPASS(kerberos-password)
    CHKADDRS
```

This keyword is used with:

- The ADDTO, REMOVE, and REPLACE commands
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- The SDT Record
- MISC3(SDT) authority to ADDTO or REMOVE records or REPLACE fields in the SDT
- MISC8(LISTSDT) authority to list realms in the SDT
- ACID(MAINTAIN) and MISC4(KERBUSER) authority

CNFAPP Keyword—Application Filter

Valid on z/OS.

Use the CNFAPP keyword to specify the application variable that acts as a filter to assign users to an ACID. Users are assigned an ACID according to the application with which they have gained access to the system. The application criteria are in addition to the digital certificate name filter defined in the corresponding TSS ADD CERTMAP command.

CNFAPP is specified on the CRITMAP command. CRITMAP is used only when MULTIID and CRITERIA are specified on a CERTMAP command indicating that criteria in addition to the subject's and/or the issuer's distinguished names are used as the filter. This criteria data is stored in the CRITMAP record on the SDT. CNFAPP is defined by CA Top Secret.

The CRITMAP command is used when MULTIID and CRITERIA are used on the CERTMAP command.

This keyword has the following format:

```
TSS ADDTO(acid) CRITMAP(recid)
                CNFAPP(application)
```

Application

Specifies the application variable that acts as a filter to assign users to an ACID. May contain an asterisk (*) for masking.

Range: Up to 8 characters.

This keyword is used with:

- The commands ADDTO, LIST, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Example: CNFAPP keyword

In this example, the special ACID name of MULTIID along with the CRITERIA name tells CA Top Secret that if the subject's and/or the issuer's distinguished name information matches, then search the CRITMAP records for a match on the application name before assigning an ACID to the user:

```
TSS ADDTO(MULTIID) CERTMAP(NYMAP2)
                   LABLCMAP('NY Dept 2 Map')
                   TRUST
                   SDNFILTR('OU=Dept2.OU=NY.OU=Sales.O=ABC Co')
                   CRITERIA(CNFAPP=&CNFAPP)

TSS ADDTO(NYDEPT2B) CRITMAP(NYCRIT2B)
                    CNFAPP(BUSINESS)

TSS ADDTO(NYDEPT2B) CRITMAP(NYCRIT2R)
                    CNFAPP(RETAIL)
```

In this example, the user whose subject's distinguished name matches the SDNFILTR is assigned the ACID NYDEPT2B or NYDEPT2R, depending upon what application was used to access the system. If access was through the BUSINESS application, NYDEPT2B is assigned to the user. If access was through the RETAIL application, NYDEPT2R is assigned.

CNFUVAR Keyword—User Filter

Valid on z/OS.

Use the CNFUVAR keyword to specify the CA Top Secret user defined criteria that acts as a filter to assign users to an ACID. The site criteria are in addition to the digital certificate name filter defined in a corresponding ADD CERTMAP command. The CRITMAP command is used when MULTIID and CRITERIA are used on the CERTMAP command. When CNFUVAR is defined on the CRITMAP command, the special ACID name, MULTIID, along with the CRITERIA name on the CERTMAP command, tell CA Top Secret that if the subject and/or the issuer name information matches, then search the CRITMAP records for a match on the variables defined by the site before assigning an ACID to the user.

This keyword has the following format:

```
TSS ADDTO(acid) CRITMAP(recid)
                CNFUVAR(site variable list)
```

site variable list

Specifies the criteria that act as a filter to assign users to an ACID. May contain an asterisk (*) for masking.

Range: Up to 8 characters.

This keyword is used with:

- The commands ADDTO, LIST, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Examples: CNFUVAR keyword

This example defines a site variable filter that assigns the accessor ID of BANKU to any users who access the system using the BANKAPP application:

```
TSS ADDTO(MULTIID) CERTMAP(BOB)
                   DCDSN(model.cert)
                   IDNFILTER('OU=')
                   SDNFILTR(('OU=')
                   CRITERIA(CNFAPPL=&CNFAPP)
                   LABLCMAP('Bobs Tellers')

TSS ADDTO(BANKU) CRITMAP(BOB)
                 CNFUVAR(CNFAPPL=BANKAPP)
```

COMMAND Keyword—Specific Commands for ACID

Valid on z/OS.

Use the COMMAND keyword to confine ACIDs to using a specific command or subset of commands available within a facility.

For purposes of LCF protection, facility entities USER77 through USER221 are not considered unique.

For information on how to allow an ACID to use all but a specific list of commands for a particular facility, see the XCOMMAND keyword.

The TSO TEST command might not be entered as an operand of the COMMAND keyword. Restricting ACIDs to the TSO TEST command allows the ACID to execute any TSO command via the TEST, LOAD, and GO sub-commands.

This keyword has the following format:

```
TSS ADDTO(acid) COMMAND(facility,(command[(G,V)]))
```

Facility

Specifies the facility commands are restricted to.

Command

Specifies the commands available to the user.

Length: 1 to 8 characters

Capacity: 1 to 30 commands

G

Indicates that the ACID can use any command beginning with the generic prefix specified.

V

Indicates that the ACID may use the command listed, but CA Top Secret will reverify the ACID's password.

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ALL Record
- MISC1(LCF) authority to ADDTO or REMOVE the COMMAND keyword from ACIDs
- MISC9(GLOBAL) authority to ADDTO or REMOVE the COMMAND resource class to the ALL record

Examples: COMMAND keyword

This example restricts a user (RABBIT) to TSO commands OPER and ACCT:

```
TSS ADDTO(RABBIT) COMMAND(TSO,(OPER,ACCT))
```

Note that the use of LCF (whether inclusive or exclusive) will not negate the controls, provided via TSO UADS, or the TSOAUTH resource of the OPER and ACCT commands. CA Top Secret will provide additional protection for these commands.

```
COMMAND(fac,(cmd(G)...)
```

This example indicates that user EJ001 can only use TSO commands prefixed by SPF:

```
TSS ADDTO(EJ001) COMMAND(TSO,(SPF(G)))
```

This example removes a command confinement from an ACID:

```
TSS REMOVE(RABBIT) COMMAND(TSO,(OPER,ACCT))
```

This example confines all users to commands within a specific facility:

```
TSS ADDTO(ALL) COMMAND(fac,(cmds...))
```

This example confines users to programs within the BATCH facility:

```
TSS ADDTO(acid) COMMAND(BATCH,(programs...))
```

This example confines users to specific SPF panels within TSO:

```
TSS ADDTO(acid) COMMAND(TSO,(SPFn,...))
```

n is equal to an SPF menu number. The VTOC menu (SPF 3.7) is entered as CMD(TSO,(SPF37)).

CONSOLE Keyword—Control Options Permissions

Valid on z/OS, z/VSE, and z/VM.

Use the CONSOLE keyword to grant or remove an ACID's ability to modify control options. For z/VM, options are modified via the TSS MODIFY command only. With z/OS, options are modified at the O/S console or via the TSS MODIFY command function.

This keyword has the following format:

```
TSS ADDTO(acid) CONSOLE
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC9(CONSOLE) authority

Examples: CONSOLE keyword

This example gives user GABVCA the ability to enter protected control options at the console, or through the TSS MODIFY command:

```
TSS ADDTO(GABVCA) CONSOLE
```

This example removes the CONSOLE attribute:

```
TSS REMOVE(GABVCA) CONSOLE
```

CPFNODE Keyword—Define CPF Node-Specific Options

Valid on z/OS.

Use the CPFNODE keyword to define CPF node-specific options to the TSS database as NDT CPFNODE elements.

This keyword has the following format:

```
TSS ADD(NDT) CPFNODE(node name)
             CPFSYSID(system smfid)
             ACTIVE(YES|NO)
             RECEIVE(ALL|NONE)
             SEND(ALL|CMD|PWD|NONE)
             GATEWAY(YES|NO)
             BROADCAST(YES|NO)
             JOURNAL(YES|NO)
             JOURNALDSN(dsname)
```

CPFNODE(*node name*)

Specifies the CPF node name.

Limits: 1-8 characters

CPFSYSID(*system smfid*)

Specifies the name of the system where this node definition applies. If not specified, the CPF node definition is used on all systems that share the security file.

ACTIVE(YES|NO)

Specifies whether the node is available to send and receive commands and passwords.

Default: YES

RECEIVE(ALL|NONE)

Specifies if the local node can receive commands from that remote node.

Default: ALL

SEND(ALL|CMD|PWD|NONE)

Specifies if the local node can send commands to other nodes. CMD specifies that only administrative command changes are sent to that node. PWD specifies that only password changes and suspensions are sent to that node.

Default: ALL

GATEWAY(YES|NO)

Specifies a node to act as a CPF gateway or CPF server for another node.

Default: NO

BROADCAST(YES|NO)

Specifies that a node is valid to receive password and command changes that are sent to all nodes using TARGET(*) or the CPFTARGET(*) control option. If NO is specified, changes are sent only when DEFNODES are associated with the acid or a node name is specified with the TARGET keyword.

Default: YES

JOURNAL(YES|NO)

Specifies if the activity for a CPF node is logged to a journal file. Specify YES to write the log data to a SYSOUT file. If SEND(NO) is also specified, nothing is written to the journal, no journal file is allocated or opened, and the status indicates that NOSPOOL is in effect. To change the status to SPOOL and open the journal, modify the SEND attribute to any option other than NO and REFRESH the NODE.

Default: YES

JOURNALDSN(*dsname*)

Specifies the data set used as a logging file when JOURNAL(YES) is specified.

The keyword is used with:

- The commands ADD, LIST, REMOVE, and REPLACE
- The NDT record
- MISC2(NDT) authority

CPFSYSID Keyword—Define CPF Global Options

Valid on z/OS.

Use the CPFSYSID keyword to define Command Propagation Facility (CPF) global options to the CA Top Secret database as NDT CPFSYSID elements.

This keyword has the following format:

```
TSS ADD(NDT) CPFSYSID(system_smfid)
                          CPFTARGET(*|LOCAL|AUTO)
                          CPFWAIT(YES|NO)
                          CPFRCVUND(YES|NO)
                          RECVCMDS(YES|NO)
                          RECVDSN(dsname)
                          CPFLISTMULT(YES|NO)
                          CPFAUTOUID(YES|NO)
                          CPFAUTOGID(YES|NO)
```

CPFSYSID(*system_smfid*)

Specifies the SMFID of the system where the CPF global options apply.

CPFTARGET(*|LOCAL|AUTO)

Specifies the default value for the TARGET keyword (for determining how to route commands). If CPF is active and a command is issued without specifying a TARGET, the product propagates the command according to the CPFTARGET setting:

LOCAL

(Default) Restricts command execution to the local node.

AUTO

Routes a command based on the DEFNODES keyword settings associated with the ACID. The user's default nodes are assigned implicitly on a TSS CREATE command function or manually by using the TSS ADDTO command function.

*

Transmits the command to all nodes defined in the CPFNODES control option (except receive-only nodes).

CPFWAIT(YES|NO)

Specifies the default value for the WAIT keyword:

YES

Specifies that CPF waits for a response from all remote nodes before processing resumes. When this processing is in effect, the CPF recovery file does *not* save transmitted commands.

NO

(Default) Specifies that CPF does *not* wait for a response from all remote nodes before processing resumes. A response is returned to the CPF journal file for that node only. The terminal keyboard is freed up as soon as the command is accepted. When this processing is in effect, the CPF recovery file saves transmitted commands. Commands that are targeted only for the local machine are *not* saved.

CPFRCVUND(YES|NO)

Specifies whether the local node can receive commands from undefined nodes.

Default: NO

RECVCMDS(YES|NO)

Specifies whether capture activity for propagated commands that are received from remote nodes and log the activity to a sysout file.

Default: YES

RECVDSN(*dsname*)

Specifies a data set that is used to hold log data when RECVCMDS(YES) is in effect.

CPFLISTMULT(YES|NO)

Specifies whether CPF allows routing LIST(ACIDS) and WHOHAS FACILITY commands to more than a single node.

Default: NO

CPFAUTOUID(YES|NO)

Specifies whether CPF transmits a TSS command with an assigned UID value (instead of the ? value).

Default: NO

CPFAUTOGID(YES|NO)

Specifies whether CPF transmits a TSS command with an assigned GID value (instead of the ? value).

Default: NO

The keyword is used with:

- The commands ADD, LIST, REMOVE, and REPLACE
- The NDT record
- MISC2(NDT) authority

More information:

DEFNODES Keyword—Default Remote Nodes (see page 181)

CRITERIA Keyword—Additional Filter Criteria

Valid on z/OS.

Use the CRITERIA keyword to define additional criteria that act as a filter to assign users to a MULTIID. CRITERIA is only used with the MULTIID special ACID on a TSS ADD CERTMAP command and indicates that variable data, in addition to distinguished name in the user's digital certificate, is used to select the ACID assigned to the user if the filter is matched.

The criteria data must be defined in the CRITMAP record to identify the ACID associated with a certificate. The CRITERIA operands act as a template and CA Top Secret automatically substitutes the actual values. Once the substitution is made, the fully expanded criteria template is used as a resource name to find a matching profile defined in the CRITMAP SDT records. One or more variable names may be specified for the CRITERIA value.

The special ACID name of MULTIID along with the CRITERIA name tells CA Top Secret that if the subject and/or the issuer name information matches, then search the CRITMAP records for a match on application name before assigning an ACID to the user.

This keyword has the following format:

```
TSS ADDTO(MULTIID)CERTMAP(recid)
                  SDNFILTR('subject-dist-name-filter')
                  IDNFILTR('issuer-dist-name-filter')
                  CRITERIA(CNFAPP=&CNFAPP.SYSID=&SYSID.variable-name1=&name1
                           .variable-name2=&name2...)
```

The application id (CNFAPP) and the system-identifier (SYSID) are defined by CA Top Secret. When a user presents a certificate to the system for identification, the identity of the application, as well as the system the user is trying to access, becomes part of the criteria. The application passes its identity to CA Top Secret, and CA Top Secret determines the system identifier. The system identifier is the four-character value specified for the SID parameter of the SMFPRMxx member of the SYS1.PARMLIB. This value is substituted for &CNFAPP and &SYSID in the criteria.

CNFAPP

Application identifier. Defined by CA Top Secret.

SYSID

System identifier. Defined by CA Top Secret.

variable name

CA Top Secret users can define their own variables as follows:

```
CRITERIA(variable name1=&name1.variable name2=&name2...)
```

The variable name is specified and the variable for which a value is substituted begins with an ampersand (&). Each criteria is separated by a period.

This keyword is used with:

- The commands ADDTO, LIST, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Examples: CRITERIA keyword

In this example, the user whose subject's distinguished name matches the SDNFILTR is assigned NYDEPT2B or NYDEPT2R, depending upon what application was used to access the system. If access was gained through the BUSINESS application, NYDEPT2B is assigned to the user. If access was gained through the RETAIL application, NYDEPT2R is assigned.

```
TSS ADDTO(MULTIID) CERTMAP(NYMAP2)
                   LABLCMAP('NY Dept 2 Map')
                   TRUST
                   SDNFILTR('OU=Dept2.OU=NY.OU=Sales.O=ABC Co')
                   CRITERIA(CNFAPP=&CNFAPP)

TSS ADDTO(NYDEPT2B) CRITMAP(NYCRIT2B)
                    CNFAPP(BUSINESS)

TSS ADDTO(NYDEPT2R) CRITMAP(NYCRIT2R)
                    CNFAPP(RETAIL)
```

CRITMAP Keyword—ACID to User Filter

Valid on z/OS.

Use the CRITMAP keyword to identify the additional criteria that act as a filter to assign an ACID to a user. The criteria filter is used only in addition to the certificate name filter defined in a corresponding TSS ADD CERTMAP command. On the TSS ADD CERTMAP command, the MULTIID special ACID and the CRITERIA keyword indicate that the criteria filter is used in addition to the certificate filter to check the identity of the user.

CRITMAP is a unique eight-byte record identifier. When criteria are defined using the TSS ADD CRITMAP command the criteria map information is stored in a CRITMAP record in the SDT on the Security File.

This keyword has the following format:

```
TSS ADDTO(acid) CRITMAP(recid)
                SYSID(system identifier)} {CNFAPP(application name)}
                CNFUVAR(site variable list)}
```

CNFAPP

The application variable defined by CA Top Secret. The value may contain an asterisk (*) for masking.

Range: Up to 8 characters

SYSID

The system identifier defined by CA Top Secret. The value may contain an asterisk (*) for masking.

Range: Up to 4 characters

CNFUVAR

Users may define their own variable.

This keyword is used with:

- The commands ADDTO, LIST, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Example: CRITMAP keyword

In this example, the special ACID name of MULTIID along with the CRITERIA name tells CA Top Secret that if the subject and/or the issuer name information matches, then search the CRITMAP records for a match on the application name before assigning an ACID to the user:

```
TSS ADDTO(MULTIID) CERTMAP(NYMAP2)
                   LABLCMAP('NY Dept 2 Map')
                   TRUST
                   SDNFILTR('OU=Dept2.OU=NY.OU=Sales.O=ABC Co')
                   CRITERIA(CNFAPP=&CNFAPP)

TSS ADDTO(NYDEPT2B) CRITMAP(NYCRIT2B)
                    CNFAPP(BUSINESS)

TSS ADDTO(NYDEPT2R) CRITMAP(NYCRIT2R)
                    CNFAPP(RETAIL)
```

DATA Keyword with ADMIN—Authority to List Information

Valid on z/OS, z/VSE, and z/VM.

Use the DATA keyword with the ADMIN command to give or remove CA Top Secret administrators the authority to list Security File information.

Note: When an ACID is given any type of DATA administration authority, DATA(NAMES) is always implied.

This keyword has the following format:

```
TSS ADMIN(acid) DATA(authority level(s))
```

This keyword can be used with:

- The commands LIST, ADMIN, and DEADMIN
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA

DATA Authority Levels

The CA Top Secret administrator can specify the authority levels:

ACIDS

Allows administrators to list all ACIDS connected to the ACID entered in the LIST command.

ADMIN

Allows administrators to list an ACID's administrative authority

ALL

Allows administrator to list everything except PASSWORD, SESSKEY, and PROFILE contents.

BASIC

Authorizes administrators to list an ACID to see the name associated with the ACID, the ACID's type, facilities he is permitted to access, profiles he is associated with, and when the ACID was created, modified, and last used.

CICS

Authorizes administrators to list values for CICS operator fields: OIDCARD, OPCLASS, OPIDENT, OPPRTY, OPTIME, SCTYKEY.

INSTDATA

Authorizes administrators to list the installation data for an ACID.

LCF

Authorizes administrators to list the commands and/or transactions that an ACID is confined to (via TRANSACTIONS/COMMAND) or restricted from (via XTRANSACTIONS/XCOMMAND).

NAMES

Allows administrators to list an ACID's name, and the associated Department or Division names.

Note: When an ACID is given any type of DATA administration authority, DATA(NAMES) is always implied.

PASSWORD

Allows administrators to display the ACID's password expiration date and interval, not the actual password.

PROFILE

Allows administrators to list contents of all profiles connected to an ACID.

RESOURCE

Authorizes administrators to list resources owned by an ACID

SESSKEY

Authorizes administrators to display the SESSKEYs associated with each LINKID in the APPCLU Record.

SMS

Authorizes administrators to list an ACID's default SMS data.

SOURCE

Authorizes administrators to list the device from which an ACID must initiate.

TSO

Authorizes administrators to list an AICD's default TSO data.

WORKATTR

Authorizes administrators to display the SYSOUT account and delivery attributes associated with a particular ACID. This includes WAACCNT, WABLDG, WADEPT, WAADDR1, WAADDR2, WAADDR3, WAADDR4, WANAME, and WAROOM.

XAUTH

Authorizes administrators to list resources which may have had PERMIT applied by an ACID within their scope, the level at which the ACID may access the resource, and the owner of the resource.

Rules

Secondary administrators and auditors can be given the ability to request all portions of a Security Record with the exception of password expiration dates, intervals, and profile contents by issuing the following ADMIN command function:

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

To give an administrator the ability to list every field of a user security record including the password expiration date and interval fields and the contents of profiles connected to the ACID, enter:

```
TSS ADMIN(acid) DATA(ALL,PASSWORD,PROFILE)
```

Examples: DATA keyword

This example gives the divisional administrator, TECHVCA, the authority to list every possible DATA field and the contents of profiles, excluding the password itself and any applicable SESSKEYs:

```
TSS ADMIN(TECHVCA) DATA(ALL,PASSWORD,PROFILE)
```

This example removes TECHVCA's authority for DATA:

```
TSS DEADMIN(TECHVCA) DATA(ALL,PASSWORD,PROFILE)
```

DATA Keyword with WHOHAS—Interpret Resource Name

Valid on z/OS and z/VM.

Use the DATA keyword with the TSS WHOHAS function to indicate if CA Top Secret interprets a resource name literally or as a mask.

When used with WHOHAS, this keyword has the following format:

```
TSS WHOHAS resclass(resname)
          DATA(MASK|NOPREFIX|LITERAL)
```

LITERAL

Specifies an exact match, regardless of the presence of masking characters. Matching permission must be a literal match both by character and by length.

MASK

Interprets masking characters for keyword(value) in the TSS WHOHAS command. Normally, masking characters are interpreted only in permissions encountered. Ignored except for resources with ATTR(MASK).

Not all resources support masking characters.

NOPREFIX

Restrict search to keyword(value) in the security record whose length matches or exceeds the length of the keyword(value) in the command. Masking characters in PERMIT is expanded but treated as a single character.

If no DATA options are specified, all matching prefixes are displayed and any masks coded on the command are not expanded.

Note: When used with the TSS WHOHAS function, the DATA keyword can only be used with the MASK, LITERAL, and NOPREFIX operands. These operands cannot be used when the DATA keyword is issued on a function other than WHOHAS.

This keyword is used with the commands ADMIN, DEADMIN, LIST, and WHOHAS.

Examples: DATA keyword

This example shows all permits:

```
TSS WHOHAS DSNAME(SYS2.)
   DATASET     = SYS2.                                  OWNER(DEPT2)
   XAUTH       = SYS2.*                                  ACID(USER0)
   ACCESS      = READ
   XAUTH       = SYS2.++                                 ACID(USER0)
   ACCESS      = READ
   XAUTH       = SYS2.PARMLIB                            ACID(USER0)
   ACCESS      = READ
   XAUTH       = SYS2.PARM                               ACID(USER0)
   ACCESS      = READ

   TSS0300I  WHOHAS   FUNCTION SUCCESSFUL
```

This example shows all permits that match exactly the resource name entered on the TSS WHOHAS command:

```
TSS WHOHAS DSNAME(SYS2.PARM) DATA(LITERAL)
    DATASET     = SYS2.                                  OWNER(DEPT2)
    XAUTH       = SYS2.PARM                               ACID(USER0)
    ACCESS      = READ

    TSS0300I  WHOHAS   FUNCTION SUCCESSFUL

TSS WHOH DSNAME(SYS2.*) DATA(LITERAL)
    DATASET     = SYS2.                                  OWNER(DEPT2)
    XAUTH       = SYS2.*                                  ACID(USER0)
    ACCESS      = READ
    TSS0300I  WHOHAS   FUNCTION SUCCESSFUL
```

This example shows all permits that match the masking pattern of the resource name entered on the TSS WHOHAS command:

```
TSS WHOHAS DSNAME(SYS2.*) DATA(MASK)
    DATASET    = SYS2.                                  OWNER(DEPT2)
    XAUTH      = SYS2.*                                  ACID(USER0)
    ACCESS     = READ
    XAUTH      = SYS2.++                                 ACID(USER0)
    ACCESS     = READ
    XAUTH      = SYS2.PARMLIB                            ACID(USER0)
    ACCESS     = READ
    XAUTH      = SYS2.PARM                               ACID(USER0)
    ACCESS     = READ
    TSS0300I  WHOHAS   FUNCTION SUCCESSFUL
```

This example shows all permits with prefixes that are equal to or longer than the resource name entered on the TSS WHOHAS command. Shorter permits are ignored

```
TSS WHOHAS DSNAME(SYS2.P) DATA(NOPREFIX)
    DATASET    = SYS2.                                  OWNER(DEPT2)
    XAUTH      = SYS2.*                                  ACID(USER0)
    ACCESS     = READ
    XAUTH      = SYS2.++                                 ACID(USER0)
    ACCESS     = READ
    XAUTH      = SYS2.PARMLIB                            ACID(USER0)
    ACCESS     = READ
    XAUTH      = SYS2.PARM                               ACID(USER0)
    ACCESS     = READ
    TSS0300I  WHOHAS   FUNCTION SUCCESSFUL
    TSS WHOH DSN(SYS2.PARM) DATA(NOPREFIX)
    DATASET    = SYS2.                                  OWNER(DEPT2
    XAUTH      = SYS2.PARMLIB                            ACID(USER0)
    ACCESS     = READ
    XAUTH      = SYS2.PARM                               ACID(USER0)
    ACCESS     = READ
    TSS0300I  WHOHAS   FUNCTION SUCCESSFUL
    TSS WHOH DSN(SYS2.PARMLIB) DATA(NOPREFIX)
    DATASET    = SYS2.                                  OWNER(DEPT2)
    XAUTH      = SYS2.PARMLIB                            ACID(USER0)
    ACCESS     = READ

    TSS0300I  WHOHAS   FUNCTION SUCCESSFUL
```

DATA Keyword with LIST—Security Record Portion

Valid on z/OS, z/VSE, and z/VM.

Use the DATA keyword with the LIST command to specify which portion of a Security Record is listed.

When used with the LIST command, this keyword has the following format:

```
TSS LIST(acid|ACID) DATA(datatype(s))
                    TYPE(USER|PROFILE|DEPARTMENT|DIVISION|ZONE
                         |SCA|LSCA|ZCA|VCA|DCA)
```

Data Types

The CA Top Secret administrator may request any or all of the following DATA types.

ACIDS

Displays all ACIDs connected to the ACID entered in the LIST function. This keyword applies to lists of division, department and profile ACIDs.

ADMIN

Displays an ACID's administrative authority.

When TSS LIST displays the administrative authorities, the designation MISC1(*ALL*), MISC3(*ALL*), MISC8(*ALL*), or MISC9(*ALL*) is used to indicate that all of the available authorities for a particular authority type in the current release have been assigned to this ACID. If one or more of the authorities available has not been assigned, a listing of the assigned authorities is provided. This example shows that three of the five MISC8 authorities have been assigned:

```
MISC8(LISTRDT,LISTSTC,LISTSDT)
```

Any restricted administration authorities designated by permits to CASECAUT entities are also displayed. This example shows that the authority to add passwords and remove asuspends has been assigned:

```
XA CASECAUT= TSSCMD.USER.ADDTO.PASSWORD                    OWNER(SYSDEPT )
   ACCESS  = UPDATE
   ADMIN BY= BY(MASTER  )    SMFID(XE05)   ON(03/01/2010)  AT(06:17:24)
XA CASECAUT= TSSCMD.USER.REMOVE.ASUSPEND                   OWNER(SYSDEPT )
   ACCESS  = UPDATE
       ADMIN BY= BY(MASTER  )    SMFID(XE05)   ON(03/01/2010)  AT(09:35:55)
```

ALL

Displays all data types except PASSWORD, PROFILE, and expiration of temporary profiles.

BASIC

(Default) Displays the ACID, the name associated with the ACID, the ACID's type, facilities that the ACID is allowed to access, profiles with which the ACID is associated, groups with which the ACID is associate including the default group, special attributes attached to the ACID, the date that the ACID was created/modified and last used, and the language preference and TZONES, if used.

Note: The AUDIT attribute is only displayed if the user performing the TSS LIST has the ACID(AUDIT) administrative attribute.

CICS

A list of values for CICS operator fields: OIDCARD, OPCLASS, OPIDENT, OPPRTY, OPTIME, SCTYKEY.

EXPIRE

Displays a list of all profile ACIDs and applicable expiration dates.

If the EXPIRE keyword is omitted, attached profiles containing expiration dates are indicated by an asterisk(*).

INSTDATA

Displays the installation data for an ACID.

LCF

Displays the commands and transactions that an ACID is confined to (via TRANSACTIONS/COMMAND) or restricted from (via XTRANS/XCOMMAND).

NAMES

Displays an ACID name, and the associated owner's name.

PASSWORD

Displays the ACID's password expiration date and interval.

PROFILES

Displays contents of all profiles connected to the ACID in the LIST function. The content of the display is based on the other DATA suboptions entered in the LIST command. For example the following command produces a list of BASIC data for PSGEDJ, and BASIC data for each profile connected to PSGEDJ:

```
TSS LIST(PSGEDJ) DATA(BASIC,PROFILE)
```

While the following command produces a list of the PSGEDJ ACID, and the name connected to that ACID. This is followed by the ACID and name of all profiles connected to PSGEDJ:

```
TSS LIST(PSGEDJ) DATA(NAMES,PROFILE)
```

Unless you specify DATA(PROFILES,NOSORT), the profiles are listed alphabetically rather than by order of processing.

RESOURCE

Displays a list of resources owned by an ACID within its scope.

Note: Use the the RESCLASS keyword with RESOURCE to limit the list of resources that are owned by the ACID. To list only data sets that are owned by DEPT1, enter:

```
TSS LIST(DEPT1) DATA(RESOURCE) RESCLASS(DATASET)
```

SOURCE

Displays devices from which an ACID must initiate.

TSO

Displays list TSO "UADS" data fields for an ACID. The TSO fields TRBA, TUPT, and TCONS are NON-Display type fields and will list as follows:

- TRBA = **NON-DISPLAY FIELD**
- TUPT = **NON-DISPLAY FIELD**
- TCONS = **NON-DISPLAY FIELD**

SESSKEY

Displays the session keys associated with designated LINKIDs in the APPCLU Record **or** PassTicket applications in the NDT Record. To view session keys, the administrator must have DATA(SESSKEY) authority and MISC2(APPCLU) or MISC1(NDT) authority depending on the keys designated.

SMS

Displays a display of fields related to DFP processing such as: SMSMGMT, SMSSTOR, SMSDATA, and SMSAPPL.

node_type

Displays all records for the node type, where node_type is one of the following:

- LDAP
- LINUX
- CPF

TERSE

Overrides the default set by the TSSCMDOPTION control option. With a TSS LIST command, TERSE suppresses the display of ADMINBY information and the ownership hierarchy beyond the owning ACID. Using TERSE enhances performance by limiting the security file I/O required to list ACID information.

VERBOSE

Overrides the default set by the TSSCMDOPTION control option. Displays all the information available to the TSS LIST command.

ENHANCED

Overrides the default set by the TSSCMDOPTION control option. Displays all ownership hierarchy information available to the TSS LIST command plus:

- MASK attribute information on a PERMIT
- ACID type for target acid in an ADMIN SCOPE listing
- RDT attribute will display PIE for Prefixed resources

WORKATTR

Displays the SYSOUT delivery and account information specified by the following attributes: WAACCNT, WABLDG, WAADDR1-4, WADEPT, WANAME, and WAROOM.

XAUTH

Displays a list of resources that may be accessed by the ACID shown in the command, the level at which the ACID may access the resource, and the owner of the resource.

Note: Use the the RESCLASS keyword with XAUTH to limit the list of resources that are cross-authorized to the ACID. To list only data sets, but not other resources that are cross-authorized to USER01, enter:

```
TSS LIST(USER01) DATA(XAUTH) RESCLASS(DATASET)
```

This keyword is used with:

- The commands ADMIN, DEADMIN, LIST, and WHOHAS
- The ACID types User, Profile, DEPARTMENT, DIVISION, ZONE, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ACIDS
- Explicit authority to LIST any or all of the TSS data types

Examples: DATA keyword

This example lists all data concerning a specific ACID:

```
TSS LIST(CATSTC1) DATA(ALL)
```

This example provides the administrator with all LIST data about the ACID or ACIDs, except password information, expiration dates of temporary profiles, associated SESSKEYs and will not list in detail any connected profiles:

```
TSS LIST(ACID|ACIDS) DATA(ALL)
```

This example obtains all LIST data including the password information (password, expiration date, interval), expiration dates of any connected profiles, as well as a detailed list of each connected profile:

```
TSS LIST(ACID|ACIDS) DATA(ALL,PASSWORD,PROFILE,EXPIRE)
```

This example lists CICS definitions for all profiles in the Payroll Department:

```
TSS LIST(ACIDS) DATA(CICS)
                DEPARTMENT(PAYROLL)
                TYPE(PROFILE)
```

This examples lists ALL properties plus IDMAP segment:

```
TSS LIST(ACID) DATA(ALL)
```

DAYS Keyword with ACIDs—Restrict Access by Day

Valid on z/OS, z/VM, and z/VSE.

Use the DAYS keyword to restrict access to a specific day(s) of the week.

This keyword has the following format:

```
TSS ADDTO(acid) resource(prefix)
                FACILITY(facility name)
                DAYS(day-list)
```

day-list

Composed of one or more of the following entries for the days of the week:

- MON—Monday
- TUE—Tuesday
- WED—Wednesday
- THU—Thursday
- FRI—Friday
- SAT—Saturday
- SUN—Sunday
- WEEKDAYS—Monday, Tuesday, Wednesday, Thursday, and Friday
- WEEKENDS—Saturday and Sunday

If the DAYS keyword is omitted, CA Top Secret allows access on every day of the week.

This keyword is used with:

- The commands ADDTO, PERMIT, REPLACE and REVOKE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ALL Record

Examples: DAYS keyword

This example gives user MICHVCA the ability to access the TSO facility on Monday and Tuesday:

```
TSS ADDTO(MICHVCA) FACILITY(TSO)
                   DAYS(MON,TUE)
```

This example changes the DAYS attribute on MICHVCA's access to the TSO facility from Monday and Tuesday to Wednesday and Thursday:

```
TSS REPLACE(MICHVCA) FACILITY(TSO)
                     DAYS(MON,TUE,WED,THU)
```

This example removes all access to the TSO facility from MICHVCA:

```
TSS REMOVE(MICHVCA) FACILITY(TSO)
```

You cannot remove the DAYS keyword from a facility, remove the facility from the ACID.

DAYS Keyword with SDT—Maintain a Calendar

Valid on z/OS, z/VSE, and z/VM.

Use the DAYS keyword to add, remove, or replace the days of the week in a calendar in the SDT Record. DAYS lets you specify what days of the week within a year are included in a CALENDAR.

When used with the SDT, this keyword has the following format:

```
TSS ADDTO(SDT) CALENDAR(cal-name)
               [YEAR(yyyy)]
               [DAYS(days)]
               [INCLUDE(mm/dd,...]
               [EXCLUDE(mm/dd,...)]
```

days

One or more of the following entries for the days of the week:

- MON—Monday
- TUE—Tuesday
- WED—Wednesday
- THU—Thursday
- FRI—Friday
- SAT—Saturday
- SUN—Sunday
- WEEKDAYS—Monday, Tuesday, Wednesday, Thursday, and Friday
- WEEKENDS—Saturday and Sunday.

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The SDT Record
- MISC3(SDT) authority

Example: DAYS keyword

This example creates a calendar for the present year with every Monday, Wednesday, Thursday and Friday enabled:

```
TSS ADDTO(SDT) CALENDAR(CAL1)
               DAYS(MON,WED,THUR,FRI)
```

DCDSN Keyword—Specify a Data Set Containing a Certificate

Valid on z/OS.

Use the DCDSN keyword to specify one of the following types of data sets:

- MVS data set that contains the digital certificate (specified as part of an ADD, CHKCERT, GENCERT, or REPLACE command)
- MVS data set that will contain the output certificates (specified as part of an EXPORT or GENREQ command).

The following requirements apply:

- The data set must be defined as physical sequential (DSORG=PS).
- The data set must be variable blocked (RECFM=VB).
- You must specify the data set name as a fully qualified name (without quotation marks).
- The data set must be cataloged.
- The certificate contained in the data set must be BER-encoded, PKCS-7 BER-encoded, or Privacy Enhanced Mail (PEM)-encoded. PEM certificates must be transported to MVS as TEXT; other formats must be transported as BINARY.

This keyword is valid only with the DIGICERT or CERTMAP keyword and has the following format:

DCDSN(*dataset_name*)

dataset_name

Specifies the name of the data set containing the digital certificate or output certificate. Specify the data set name as a fully qualified name (without quotation marks).

Length: Up to 44 characters

This keyword is used with:

- The commands ADDTO, REPLACE, CHKCERT, GENCERT, GENREQ, and EXPORT
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority
- MISC4(CERTSITE) authority for CERTSITE ACID
- MISC4(CERTAUTH) authority for CERTAUTH ACID

Example: Add a Digital Certificate

This example adds digital certificate CERT0001 to USER1. The certificate is contained in z/OS data set USER1.CERT.DATA.

```
TSS ADDTO(USER1) DIGICERT(CERT0001)
DCDSN(USER1.CERT.DATA)
```

Example: Generate a Self-Signed Certificate Authority Certificate

This example generates a self-signed certificate authority certificate from a PKCS#10 request. The generated certificate is placed in data set CERTREQ.REQ.

```
TSS GENCERT(CERTAUTH) DIGICERT(Account)
DCDSN(CERTREQ.REQ) NBDATE(05/01/02) NADATE(05/01/03)
```

DCENCRY Keyword—Encryption Key Value

Valid on z/OS and z/VM.

Use the DCENCRY keyword in conjunction with KEYSMSTR to provide a key name to encrypt and decrypt passwords.

On z/OS to have the SMB server use encrypted password processing, add the entry DCE.PASSWORD.KEY to the SDT KEYSMSTR record.

To use the EIM/PROXY feature, add the KEYSMSTR entry LDAP.BINDPW.KEY before entering a PRXBINDPW.

This keyword has the following format:

```
TSS ADDTO(SDT) KEYSMSTR(XXX.XXXXXX.XXX)
               DCENCRY(CCCCCCCCCCCCCCCC)
               [KEYMASK|KEYENCRY]
```

CCCCCCCCCCCCCCCC

Hexadecimal encryption key value.

Length: 16 characters

KEYMASK

(Default) Indicates that the DCENCRY key is used to mask the user's DCE password when it is stored in the DCEKEY field of the user's acid record.

KEYENCRY

Indicates that the DCENCRY key is used to encrypt the user's DCE password when it is stored in the user's acid record.

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The SDT records exclusively
- MSCA authority

Examples: DCENCRY keyword

This example defines the string C1C2C3C4C5C6C7C8 as the encryption key value for the LDAP PROXY BINDPW key:

```
TSS ADD(SDT) KEYSMSTR(LDAP.BINDPW.KEY)
             DCENCRY(C1C2C3C4C5C6C7C8)
```

This example lists the KEYSMSTR record:

```
TSS LIST(SDT) KEYSMSTR(LDAP.BINDPW.KEY)
```

This example deletes the KEYSMSTR from the SDT:

```
TSS DELETE(SDT) KEYSMSTR(LDAP.BINDPW.KEY)
```

DCLASS Keyword—Data Classification Class Definition

Valid on z/OS.

Use the DCLASS keyword to define a Data Classification class name. This keyword is part the of DATACLAS tracking record that helps determine which data (files, data sets, resources) pertains to which regulation. This record resides in the VSAM file.

The DCLASS keyword must be specified when administering the DATACLAS record.

The name can contain letters, numbers, or special characters. If the DCLASS name contains blanks, place quotation marks around the name.

Keyword length

One to 32 characters.

Capacity of list

One data classification class per command.

This keyword is used with:

- The commands LIST, ADDTO, and REMOVE
- The DATACLAS record only
- MISC5(DCLADMIN) authority
- MISC5(DCLLIST) authority to LIST data classes in the DATACLAS record

Examples: DCLASS Keyword

This example adds a data classification record under the classification HIPAA:

```
TSS ADDTO(DATACLAS) DCLASS(HIPAA)
                    RCLASS(DATASET)
                    RESNAME(HIPAA.RECORDS)
                    OWNER1(CLSOWNER)
```

This example lists all data classification records in the HIPAA classification:

```
TSS LIST(DATACLAS) DCLASS(HIPAA)
```

DEFACC Keyword—Resource Default Access

Valid on z/OS, z/VSE, and z/VM.

Use the DEFACC keyword to assign the default access used on a TSS PERMIT for a resource that has been added to the RDT Record.

This keyword has the following format:

```
TSS ADDTO(RDT) RESCLASS(resource Type)
               RESCODE(hex code)
               [DEFACC(access list)]
```

If not specified, the default access is NONE

Note: The access level specified by DEFACC must match the applicable access levels indicated by the ACLST entries for that resource. If they do not match or, if no ACLST was specified, you will receive a TSS0282E error message.

This keyword is used with:

- The commands REPLACE and ADDTO
- The RDT ACID only
- MISC1(RDT) authority

Access Levels

The predefined CA Top Secret access levels, with their associated hexadecimal values are:

ALL=FFFF	AUTOLOG=4000	BLP=8000
BROWSE=0200	COLLECT=0002	CONTROL=0400
CREATE=1000	DELETE=1000	FEOV=0200
FETCH=8000	FIND=1000	GRPLOGON=1000
LOGON=8000	MREAD=4400	MWRITE=2400
MULTI=0400	NOCREATE=0100	NONE=0000
NONSHR=2000	PURGE=0100	READ=4000

REPL=0800	SCRTCH=0800	SHR=4000
SUROGATE=2000	UPDATE=8000	WRITE=2000

Examples: DEFACC keyword

This example creates a user-defined resource class. If the access level is not known to CA Top Secret, specify the hex value in the DEFACC as well as the ACLST fields:

```
TSS ADDTO(RDT) RESCLASS(NEWRES)
               RESCODE(001)
               ACLST(ALLOW=4000)
               DEFACC(ALLOW=4000)
```

This example establishes a TSS PERMIT default access level of READ to the #PRODUCT resource:

```
TSS ADDTO(RDT) RESCLASS(#PRODUCT)
               RESCODE(002)
               DEFACC(READ)
```

Now, when any #PRODUCT resource is permitted to a user and the ACCESS keyword is omitted from the PERMIT, a default access level of READ is used.

DEFAULT Keyword—Key Ring Default Certificate

Valid on z/OS.

Use the DEFAULT keyword to specify that the digital certificate being added to a key ring is the default certificate for the key ring. Only one certificate within the key ring can be the default. If a default already exists, its DEFAULT status is removed, and the specified certificate becomes the default certificate.

This keyword has the following format:

```
TSS ADDTO(acid) KEYRING(8-byte name)
               [LABLRING(237-byte ring name)]
                \{RINGDATA(acid,digicert)\}
                \{RINGDATA(CERTSITE,digicert)\}
                \{RINGDATA(CERTAUTH,digicert)\}
               [DEFAULT]
               [USAGE(PERSONAL|CERTSITE|CERTAUTH)]
```

This keyword is used with:

- The commands ADDTO, P11TOKEN, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority for users
- MISC4(CERTUSER) authority for user IDs
- MISC4(CERTSITE) authority for CERTSITE ACIDs
- MISC4(CERTAUTH) authority for CERTAUTH ACIDs

Example: DEFAULT keyword

This example indicates that this is the default certificate for a key ring:

```
TSS ADDTO(USER1) KEYRING(CASRING)
                 RINGDATA(USERA,USERACER)
                 DEFAULT
```

DEFNODES Keyword—Default Remote Nodes

Valid on z/OS, z/VSE, and z/VM.

Use the DEFNODES keyword to maintain a list of default routing nodes for an individual ACID. The list is consulted when:

- The CPFTARGET control option is set to AUTO and no TARGET is specified when the command is issued
- The CPFTARGET control option is set to * and TARGET(SELECT) is specified when the command is issued

The DEFNODES keyword is only used when the list itself is being designated (through the initial TSS CREATE for the ACID or through a subsequent TSS ADDTO) or updated. Otherwise, the list is consulted by default when *both* of the above conditions apply.

If CPFTARGET is set to AUTO and no TARGET is specified but the ACID has no DEFNODES, the command propagates to those nodes identified by the CPFNODES control option.

The DEFNODES list can be assigned to an ACID from the initial CREATE statement or it can be added later.

This keyword has the following format:

```
TSS ADDTO(acid) DEFNODES(oper,oper,...)
```

You can specify up to five nodes per command.

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(DEFNODES) authority

Examples: DEFNODES keyword

This example defines the default routing nodes of NYC, CHI and DET for the USER01 ACID:

```
TSS ADDTO(USER01) DEFNODES(NYC,CHI,DET)
```

If a command is issued against USER01 and:

- CPFTARGET is set to AUTO and TARGET is specified on the command, the command is propagated to the NYC, CHI, and DET nodes
- CPFTARGET is set to * and TARGET(SELECT) is specified on the command and the CPFNODES control option is CPFNODES(ATL,NYC(NB),CHI(NB),DET(NB)), the command is sent to the NYC, CHI and DET nodes

This example completely removes the list:

```
TSS REMOVE(USER01) DEFNODES(NYC,CHI,DET)
```

This example defines a new user for the Accounting Department. The new user requires an initial DEFNODES list that includes New York, Chicago, and Detroit:

```
TSS CREATE(ACCT01) NAME('ACCT USER')
                   FACILITY(VM,TSO)
                   PASSWORD(xxxx)
                   TYPE(USER)
                   DEFNODES(NYC,CHI,DET)
```

The administrator has the option of using a Model ACID, which contains all the default access rights and restrictions—including DEFNODES—for all users in the Accounting Department. In this case, specify the USING keyword with the TSS CREATE.

This example adds DEFNODES Baltimore and Atlanta to the user:

```
TSS ADDTO(USER01) DEFNODES(BAL,ATL)
```

The list for USER01 now consists of NYC, CHI, DET, BAL, ATL.

This example removes Houston from the list:

```
TSS REMOVE(USER01) DEFNODES(HOU)
```

This example completely replaces the current list with a new list consisting of Houston and Boston:

```
TSS REPLACE(USER01) DEFNODES(HOU,BOS)
```

DEFTKTLF Keyword—Default Ticket Life

Valid on z/OS.

Use the DEFTKTLF keyword to specify the default ticket life in the KERBDFLT REALM record in the SDT.

This keyword has the following format:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME('kerberos-realm-name')
               MINTKTLF(min-ticket-life)
               MAXTKTLF(max-ticket-life)
               DEFTKTLF(nnnnnnnnnn)
               KERBPASS(kerberos-password)
               CHKADDRS
```

DEFTKTLF

Defines the default ticket life in seconds. This keyword is only applicable when defining the KERBDFLT realm record (not foreign realms). If DEFTKTLF is specified, then MINTKTLF and MAXTKTLF must also be specified.

Range: 1 to 2147483647 seconds

Default: 300 (5 minutes)

This keyword is used with:

- The commands ADDTO, DELETE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- MISC3(SDT) authority to ADDTO or REMOVE records or fields in the SDT
- MISC8(LISTSDT) authority to list KERBNAME in the SDT

Example: DEFTKTLF keyword

This example indicates a default ticket life of ten hours in the default REALM SDT record:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME('LOCAL.CA.COM')
               MINTKTLF(30)
               MAXTKTLF(86400)
               DEFTKTLF(36000)
               KERBPASS(CHILDREN)
               CHKADDRS
```

DEPARTMENT Keyword—Department ACID Assignment

Valid on z/OS, z/VSE, and z/VM.

Use the DEPARTMENT keyword to:

- Assign a new ACID to a department
- List data about ACIDs in a department

When used with CREATE, this keyword has the following format:

```
TSS CREATE(user|profile|DCA acid) NAME('name')
                                  FACILITY(facname)
                                  PASSWORD(password)
                                  DEPARTMENT(acid)
```

TYPE(USER) and TYPE(PROFILE) ACIDs must be created in a department.

Since a DCA can only create ACIDs in his own department, the new ACID is assigned to the DCA's department. CA Top Secret does not accept the DEPARTMENT keyword when entered by a DCA, even if the DCA specifies his own department ACID. The DEPARTMENT keyword is only required when the person entering the command is not a DCA.

When used with LIST, this keyword has the following format:

```
TSS LIST(ACID|ACIDS) DATA(datatype(s))
                     TYPE(USER|PROFILE|DCA)
                     DEPARTMENT(acid)
```

CA Top Secret only displays data concerning ACIDs within the administrator's scope. An MSCA or an authorized SCA could obtain LIST data for the entire site. A VCA may list data for his division and all subordinate departments. A DCA may not specify the DEPARTMENT keyword.

This keyword is used with:

- The commands LIST, MOVE, and CREATE
- The ACID types User, Profile, and DCA

Examples: DEPARTMENT keyword

This example (entered by a DCA) creates a new TSO user for the TECH Department:

```
TSS CREATE(TECH10) NAME('TECH USER')
                   FACILITY(TSO)
                   PASSWORD(XXXX,15,EXP)
                   TYPE(USER)
```

When TECH10 signs on using his password, he is automatically prompted for a new password which expires every 15 days

This example is the same entry from an administrator other than the DCA:

```
TSS CREATE(TECH10) NAME('TECH USER')
                   FACILITY(TSO)
                   PASSWORD(XXXX,15,EXP)
                   DEPARTMENT(TECHDEP)
                   TYPE(USER)
```

This example lists the names of all users in the TECH Department:

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

DESCRIPT Keyword—List Description

Valid on z/OS, z/VSE, and z/VM.

Use the DESCRIPT keyword to:

- Specify a description that displays with the LIST command function. Administrators can add, remove, or replace a user's description field in any of the six record types in the SDT Record.
- Add a description to an existing calendar.
- Add comments to a DATACLAS record (RCLASS, RESNAME, OWNER1, and ONAME1). For the format and example of this option, see Example: DESCRIPT keyword.

This keyword has the following format:

```
TSS ADDTO(SDT) [CALENDAR(cal-name)]
               DESCRIPT(descript-name)
               [MAP(map-name)]
               [MASK(mask-name)]
               [RLP(record-name)]
               [SELECT(select-name)]
               [TIMEREC(time-name)]
```

descript-name

User description field that permits the administrator to apply a logical name to this particular record. If the description field contains blanks, enclose it in single quotes.

Length: Up to 32 characters

This keyword is used with:

- The commands ADDTO and REPLACE
- The SDT Record exclusively
- MISC3(SDT) authority

More information:

Examples: RESCLASS keyword (see page 426)
Examples: OWNER1 Keyword (see page 379)
Example: ONAME1 Keyword (see page 373)
Examples: RESNAME Keyword (see page 430)

Example: DESCRIPT keyword

This example creates a calendar for the present year with a user-description field called PAYROLL CALENDAR:

```
TSS ADDTO(SDT) CALENDAR(CAL1)
               DESCRIPT('PAYROLL CALENDAR')
```

This example adds a resource class DATASET to the HIPAA DATACLAS record:

```
TSS ADDTO(DATACLAS) DCLASS(HIPAA)
                    RCLASS(DATASET)
                    RESNAME(HIPAA.RECORD)
                    OWNER1(HIPOWNER)
                    ONAME1(HIPOWNER OWNS ALL HIPAA DATASETS)
                    DESCRIPT('HIPAA')
```

DFLTGRP Keyword—ACID Default Group

Valid on z/OS and z/VM.

Use the DFLTGRP keyword to assign a default group to an ACID operating under OpenEdition MVS. Every user type ACID in a z/OS environment must be assigned a default group.

This keyword has the following format:

```
TSS ADDTO(acid) DFLTGRP(groupname)
```

This keyword is used with:

- The commands ADDTO, REMOVE, REPLACE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: DFLTGRP keyword

This example defines the default group of OS390GRP to ACID OS390U2:

```
TSS ADDTO(OS390U2) DFLTGRP(OS390GRP)
```

This example removes the OS390GRP group from ACID OS390U2:

```
TSS REMOVE(OS390U2) DFLTGRP(OS390GRP)
```

DFLTSLBL Keyword—Default SECLABEL Association

Valid on z/OS.

Use the DFLTSLBL keyword to associate a default MLS SECLABEL label with a selected ACID.

This keyword has the following format:

```
TSS ADDTO|REMOVE|REPLACE(acid) SECLABEL(label1[,...,label5])
                    [DFLTSLBL(labeld)]
```

acid

Specifiers the accessor-id being modified.

label1,...label5

Specifies a list of previously defined SECLABEL labels in the MLS record.

Capacity: Up to 5

labeld

Specifies the default SECLABEL assigned when the process does not otherwise assign a SECLABEL. The default SECLABEL must be one of the MLS labels assigned to the acid.

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC5(MLS) authority

Example: DFLTSLBL keyword

This example associates the ACID KONG with the DFLTSLBL FAYWRAY.

```
TSS ADDTO(KONG) SECLABEL(SECHIGH,FAYWRAY)
                DFLTSLBL(FAYWRAY)
```

DIGICERT Keyword—Identify a Digital Certificate

Valid on z/OS.

Use the DIGICERT keyword to identify a digital certificate.

Note: You must use this keyword with ADDTO, ROLLOVER, GENCERT, DIGICERT, and REKEY functions with digital certificates.

When adding a DIGICERT using DCDSN that contains a PKCS12 package, the first certificate is added to the user with the signing certificates. The certificates are added to CERTAUTH ACID with label-generated names of AUTOXXX between 0 and 1000.

If an error occurs during the add function, certificates added to CERTAUTH are not backed out.

This keyword has the following format:

```
TSS COMMAND(acid) DIGICERT(name)
                  [DCDSN(dsname)]
```

name

Specifies a case-sensitive character ID that identifies the digital certificate to an ACID.

Length: 1 to 8 characters

This keyword is used with:

- The commands ADDTO, EXPORT, GENCERT, GENREQ, LIST, REKEY, REMOVE, RENEW, REPLACE, and ROLLOVER
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTUSER) authority

Administrators without ACID(MAINTAIN) and MISC4(CERTUSER) authority need:

- UPDATE access to TSSCMD.CERTUSER.ADDTO in the CASECAUT resource class to ADD a certificate to a user ACID
- UPDATE access to TSSCMD.CERTUSER.REMOVE in the CASECAUT resource class to REMOVE a certificate from a user ACID
- UPDATE access to TSSCMD.CERTSITE.ADDTO in the CASECAUT resource class to ADD a site certificate
- UPDATE access to TSSCMD.CERTSITE.REMOVE in the CASECAUT resource class to REMOVE a site certificate
- UPDATE access to TSSCMD.CERTAUTH.ADDTO in the CASECAUT resource class to REMOVE a certificate-authority certificate
- UPDATE access to TSSCMD.CERTAUTH.REMOVE in the CASECAUT resource class to REMOVE a certificate-authority certificate

Example: Add a Certificate

This example adds a digital certificate with the name CERT0001 to USER1:

```
TSS ADDTO(USER1) DIGICERT(CERT0001)
                 DCDSN(USER1.CERT.DATA)
```

Example: List Certificates on a System

This example lists all the digital certificates on a system:

```
TSS LIST(ACIDS) DIGICERT(ALL)
```

Example: Remove a Certificate

This example removes the digital certificate from the Security File:

```
TSS REMOVE(USER1) DIGICERT(CERT0001)
```

Example: Roll Over a Certificate

This example rolls over a certificate with a REKEY certificate:

```
TSS ROLLOVER(myacid) DIGICERT(TEST)
                    NEWDIGIC(NEWTEST)
```

Example: Generate a Certificate Named TEST

This example generates a certificate named TEST within the acid record:

```
TSS GENCERT(myacid) DIGICERT(TEST)
```

Example: Generate a Certificate Based on Another Certificate

This example generates the new certificate NEWDIGIC based on a certificate in the DIGICERT field:

```
TSS REKEY(myacid) DIGICERT(TEST)
                  NEWDIGIC(NEWTEST)
```

DISPLAY Keyword—FDT Display Field

Valid on z/OS, z/VM, and z/VSE.

Use the DISPLAY keyword:

- To define, change, or list the display value for the field in the FDT Record. If the DISPLAY value in not given, it defaults to the value specified for FDTNAME.
- To identify the field name associated with a given display value.

When used with ADDTO, the command has the following format:

```
TSS ADDTO(FDT) DISPLAY(fieldname)
               FDTNAME(field name)
               FDTCODE(hex code)
```

fieldname

Specifies the display value for the field in the FDT Record.

Length: 1 to 11 characters

When used with LIST, this keyword has the following format:

```
TSS LIST(FDT) DISPLAY('display value')
```

This keyword is used with:

- The commands ADDTO, REPLACE, and LIST
- The ACID type FDT
- The FDT record only
- MISC1(RDT) authority

Examples: DISPLAY Keyword

This example adds a new field called $PERS to the FDT Record with a DISPLAY value of EMPLOY:

```
TSS ADDTO(FDT) FDTNAME($PERS)
               FDTCODE(1E)
               SEGMENT(PERSDEPT)
               MAXLEN(11)
               DISPLAY('EMPLOY')
```

This example finds the field name associated with the LOCKTIME= value:

```
TSS LIST(FDT) DISPLAY('LOCKTIME')
```

DIVISION Keyword—Division Data

Valid on z/OS, z/VSE, and z/VM.

Use the DIVISION keyword to:

- List data about ACIDs in a specific division
- Assign an ACID to a division

When used with LIST, this keyword has the following format:

```
TSS LIST(ACID|ACIDS) DATA(dataType(s))
                     TYPE(acidtype)
                     DIVISION(acid)
```

When used with CREATE, this keyword has the following format:

```
TSS CREATE(dept acid|VCA acid) TYPE(DEPARMENT|VCA)
                               NAME('VCA or Dept name')
                               DIVISION(acid)
```

TYPE(USER) and TYPE(PROFILE) ACIDs must be created in a department.

This keyword is used with:

- The commands CREATE, MOVE, and LIST
- The ACID types Division, Zone, VCA, ZCA, LSCA, and ACIDS

CA Top Secret only displays data concerning ACIDs within their administrator's scope. An MSCA or an authorized SCA could obtain LIST data for the entire site. A VCA cannot specify the DIVISION keyword.

Examples: DIVISION keyword

This example lists the names of all users in the TECH Division:

```
TSS LIST(ACIDS) DATA(NAMES)
                DIVISION(TECH)
```

This example creates a new Accounting Department placed in the Parts Division:

```
TSS CREATE(ACCT01) NAME('ACCT DEPARMENT')
                   TYPE(DEPARMENT)
                   DIVISION(PARTDIV)
```

DSA—Generate Keys with DSA

Valid on z/OS.

Use the DSA keyword to specify that the key pair is generated using the Digital Signature Algorithm instead of the RSA algorithm. The DSA algorithm creates key pairs that can only be used to sign data. The RSA algorithm creates key pairs that can be used to sign data and to encrypt data. This parameter cannot be used with the ICSF, PCICC, NISTECC, or BPECC parameters. When specifying the DSA parameter, the KEYSIZE parameter can be as high as 2048.

This keyword has the following format:

```
TSS GENCERT(acid) DIGICERT( 8–byte name)
                  SUBJECTN(subject–name)
                  [LABLCERT( label name)]
                  [DSA]
                  [TRUST|NOTRUST]
                  KEYSIZE(512-2048)
```

This keyword is used with:

- The GENCERT and REKEY commands
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority for users
- MISC4(CERTSITE) for authority CERTSITE ACIDs
- MISC4(CERTAUTH) for authority CERTAUTH ACIDs

Examples: DSA keyword

This example uses DSA to generate a key pair:

```
TSS GENCERT(user1) DIGICERT(cert0001)
                   SUBJECTN(CN=user1certificate)
                   DSA
                   KEYSIZE(512)
```

DUFUPD Keyword—INSTDATA and Security Record Updates

Valid on z/OS, z/VM, and z/VSE.

Use the DUFUPD keyword to add or remove an ACID's DUFUPD. DUFUPD enables an ACID to use the CA Top Secret Application Interface to update the installation data (INSTDATA) or field data from a Security Record. DUFUPD is a component of the CA Top Secret Dynamic Update Facility (DUF).

This keyword has the following format:

```
TSS ADDTO(acid) DUFUPD
```

The DUFUPD keyword is only applicable when installation or field data used for customization is entered into the Security File.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE and WHOHAS
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC1(INSTDATA) authority

Examples: DUFUPD keyword

This example gives user PRS001 the ability to update installation data:

```
TSS ADDTO(PRS001) DUFUPD
```

This example removes the DUFUPD attribute:

```
TSS REMOVE(PRS001) DUFUPD
```

This example uses DUFXTR and DUFUPD together to provide full use of the Dynamic Update Facility:

```
TSS ADDTO(acid) DUFXTR DUFUPD
```

DUFXTR Keyword—INSTDATA and Security File Data Extraction

Valid on z/OS, z/VM, and z/VSE.

Use the DUFXTR keyword to add or remove an ACID's DUFXTR attribute. DUFXTR enables an ACID to use a RACROUTE REQUEST=AUTH (RACHECK) macro or the CA Top Secret Application Interface to extract installation data (INSTDATA) or field data from a Security File. DUFXTR is a component of the CA Top Secret Dynamic Update Facility (DUF).

This keyword has the following format:

```
TSS ADDTO(acid) DUFXTR
```

The DUFXTR keyword is only applicable when installation or field data used for customization is entered into the Security File.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE and WHOHAS
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC1(INSTDATA)

Examples: DUFXTR keyword

This example gives user PRS001 the ability to extract installation data:

```
TSS ADDTO(PRS001) DUFXTR
```

This example removes the DUFXTR attribute:

```
TSS REMOVE(PRS001) DUFXTR
```

This example uses DUFXTR and DUFUPD together to provide full use of the Dynamic Update Facility:

```
TSS ADDTO(acid) DUFXTR DUFUPD
```

EIMDOMAIN Keyword—EIM Distinguished Name

Valid on z/OS.

Use the EIMDOMAIN keyword to create an EIM profile on the SDT.

The EIMDOMAIN keyword is the distinguished name of an EIM domain. The field can be mixed case and up to 1023 characters in length. EIMDOMAIN is used in conjunction with the following keywords to create an EIM profile on the SDT: EIMPROF, EIMOPTION, EIMLOCREG, PRXLDAPHST, PRXBINDDN, and PRXBINDPW.

This keyword has the following format:

```
TSS ADDTO(SDT) EIMPROF(eim_profile_name)
               EIMOPTION(ON|OFF)
               EIMDOMAIN(name)
               EIMLOCREG(name)
               PRXLDAPHST(name)
               PRXBINDDN(name)
               PRXBINDPW(password)
```

This keyword is used with:

- MISC3(SDT) authority
- The SDT Record exclusively

Example: EIMDOMAIN keyword

This example creates an EIMPROF record and adds it to the SDT:

```
TSS ADDTO(SDT) EIMPROF(eim1)
               EIMOPTION(on)
               EIMDOMAIN('xxx_eimDomainName=EIM
                         Domain,o=Company,
                         st=State,c=Country')
               EIMLOCREG('REGISTRY XE')
               PRXLDAPHST('LDAP HOST')
               PRXBINDDN('ldap://domain')
               PRXBINDPW(bind password)
```

EIMLOCREG Keyword—EIM Local Registry Name

Valid on z/OS

Use the EIMLOCREG keyword to create an EIM profile on the SDT.

When used with ADDTO or REPLACE, this keyword has the following format:

```
TSS ADDTO(SDT) EIMPROF(eim–profile–name)
               EIMOPTION(ON|OFF)
               EIMDOMAIN(name)
               EIMLOCREG(name)
               PRXLDAPHST(name)
               PRXBINDDN(name)
               PRXBINDPW(password)
```

EIMLOCREG

The name of the local registry. The field can be mixed case.

Length: Up to 255 characters

When used with DELETE, this command has the following format:

```
TSS DELETE(SDT) EIMPROF(eim–profile–name)
```

This keyword is used with:

- The SDT Record exclusively
- MISC3(SDT) authority
- The following keywords in creating an EIM profile on the SDT: EIMPROF, EIMOPTION, EIMDOMAIN, PRXLDAPHST, PRXBINDDN, and PRXBINDPW.

Example: EIMLOCREG keyword

This example creates an EIMPROF record and add it to the SDT:

```
TSS ADD(SDT) EIMPROF(eim1)
             EIMOPTION(on)
             EIMDOMAIN('xxx–eimDomainName=EIM Domain,
                       o=Company,st=State,c=Country')
             EIMLOCREG('REGISTRY XE')
             PRXLDAPHST('LDAP HOST')
             PRXBINDDN('ldap://domain')
             PRXBINDPW(bind password)
```

EIMOPTION Keyword—EIM Connection Permission

Valid on z/OS .

Use the EIMOPTION keyword to specify whether or not new connections may be established with the specified EIM domain.

When used with ADDTO or REPLACE, this keyword has the following format:

```
TSS ADDTO(SDT) EIMPROF(eim_profile_name)
               EIMOPTION(ON|OFF)
               EIMDOMAIN(name)
               EIMLOCREG(name)
               PRXLDAPHST(name)
               PRXBINDDN(name)
               PRXBINDPW(password)
```

When used with DELETE, this keyword has the following format

```
TSS DELETE(SDT) EIMPROF(eim_profile_name)
                EIMOPTION(ON|OFF)
                EIMDOMAIN(name)
                EIMLOCREG(name)
                PRXLDAPHST(name)
                PRXBINDDN(name)
                PRXBINDPW(password)
```

This keyword is used with:

- The following keywords in creating an EIM profile on the SDT: EIMPROF, EIMDOMAIN, EIMLOCREG, PRXLDAPHST, PRXBINDDN, and PRXBINDPW
- The SDT Record exclusively
- MISC3(SDT) authority

EIMPROF Keyword—Supply an EIM Profile Name for an ACID

Valid on z/OS.

Use the EIMPROF keyword to specify the EIM profile name for EIM and PROXY user information of the ACID.

EIM is an infrastructure technology for managing multiple identities in an enterprise. Maintaining relationships that map identities allows administrators and applications to use these relationships to access identities.

EIM lets an organization create a unique identity for an individual (or resource) and relate the identity to all other representations of the given entity (individual or resource) within the organization. EIM provides a means to manage multiple user registries on multiple platforms. EIM stores information about the relationships between other identities and allows EIM-enabled applications to map from one identity to another identity.

EIM uses information that is stored in a security server to connect to an EIM domain. Additionally, the security server may participate in an EIM domain as a user registry.

EIM-enabled applications can perform EIM lookup operations to get information from an EIM domain. An application supplies lookup information (such as the userid of a user in a given registry), and the retrieved information could be an associated userid in a target registry.

Note: To enter information into the default EIM record, use EIMPROF(EIMDEF). To enter information into the default PROXY record, use EIMPROF(PROXYDEF).

When used with ADDTO and REPLACE, the EIMPROF keyword has the following format:

```
TSS ADDTO(SDT) EIMPROF(eim_profile_name)
               EIMOPTION(ON|OFF)
               EIMDOMAIN(name)
               EIMLOCREG(name)
               PRXLDAPHST(name)
               PRXBINDDN(name)
               PRXBINDPW(password)
```

When used with DELETE, the EIMPROF keyword has the following format:

```
TSS DELETE(SDT) EIMPROF(eim_profile_name)
```

When used with LIST, the EIMPROF keyword has the following format:

```
TSS LIST(SDT) EIMPROF(eim_profile_name)
```

When used with REMOVE, the EIMPROF keyword has the following format:

```
TSS REMOVE(acid) EIMPROF
```

Add the EIMPROF keyword to an ACID with the following format:

```
TSS ADDTO(acid) EIMPROF(eim_profile_name)
```

This keyword is used with:

- The SDT Record exclusively
- MISC3(SDT) authority
- The keywords EIMOPTION, EIMDOMAIN, EIMLOCREG, PRXLDAPHST, PRXBINDDN, and PRXBINDPW.

ENCRYPT Keyword—Encryption Level

Valid on z/OS.

Use the ENCRYPT keyword to enable or disable levels of encryption. The levels supported are:

ENCRYPT

Configuration ENCTYPE

- des-cbs-crc
- des-cbc-md4
- des-cbc-md5

DESD

Configuration ENCTYPE double DES.

DES3

Configuration ENCTYPE:

- des3-cbc-sha1
- des-cbc-crc

AES128

Configuration ENCTYPE:

- aes128-cts-hmac-sha1-96

AES256

Configuration ENCTYPE:

- aes256-cts-hmac-sha1-96

Corresponding to each level of encryption in the security environment, there must be a corresponding level in the Kerberos configuration file. See the IBM documentation on the Security Server Network Authentication Service to assure that your configuration file corresponds to your security encryption specification.

The encryption levels of mutually defined systems in a TCP/IP network must specify equal encryption levels for handshake:

LOCAL REALM A KERBPASS: X	LOCAL REALM B KERBPASS: Y
FOREIGN REALM B KERBPASS: Y	FOREIGN REALM A KERBPASS: X

For z/OS 1.9 and above, KERBLVL is ignored, all levels are supported.

When used with REALM, this keyword has the following format:

```
TSS ADD(SDT) REALM(KERBDFLT|foreign_realm)
             REALMNAME(realmname)
             ENCRYPT('[DES|NODES]
                      [DES3|NODES3]
                      [DESD|NODESD]
                      [AES128|NOAES128]
                      [AES256|NOAES256]')
             KERBPASS(password)
             CHKADDRS
```

Note: The CHKADDRS keyword is used only with KERBDFLT realm record, not foreign realms.

When used with Kerberos, this keyword has the following format:

```
TSS ADD(acid) KERBNAME(kerbname)
              ENCRYPT('[DES|NODES])
                      [DES3|NODES3]
                      [DESD|NODESD]
                      [AES128|NOAES128]
                      [AES256|NOAES256]')
```

ENCRYPT

- DES 56 bit encryption
- DESD double DES encryption
- DES3 168 bit encryption
- AES128 AES 128 bit encryption
- AES256 AES 256 bit encryption

Default: DES DES3 DESD AES128 AES256

The keyword is used with:

- The commands ADDTO and REPLACE
- The SDT Record and on the definition of Kerberos Principal Users
- MISC3(SDT) authority to ADDTO or REMOVE records or REPLACE fields in the SDT
- MISC8(LISTSDT) authority to list realms in the SDT

Examples: ENCRYPT keyword

This example enables the encryption levels DES and DES3 in the local REALM:

```
TSS ADD(SDT) REALM(KERBDFLT)
             REALMNAME(HYPOTHETICAL.CA.COM)
             KERBPASS(THET1CAL)
             ENCRYPT('NODESD')
             CHKADDRS
```

This example limits the encryption for a particular user to DES encryption only:

```
TSS ADD(KRBPEON) KERBNAME(KRBPEON)
                 ENCRYPT('NODESD NODES3')
```

This example defines a local realm with an explicit encryption level:

```
TSS ADD(SDT) REALM(KERBDFLT)
             REALMNAME('BOTTOMFEEDER.CARP.COM')
             ENCRYPT('DES DES3 NODESD')
             KERBPASS(GZORNPLT)
             CHKADDRS
```

This example defines ACID TESTER with a lesser encryption level:

```
TSS ADD(TESTER) KERBNAME(TESTER)
                ENCRYPT('DES NODES3 NODESD')
```

ENCRYPT Keyword—Encryption Level Override

Valid on z/OS.

Use the ENCRYPT keyword to override the value of ENCRYPT for the local REALM and set the certificate encryption level available to a particular user.

For z/OS 1.8 and below:

- When Control Option KERBLVL(0) is set, only DES is supported.
- When Control Option KERBLVL(1) is set, all levels are supported.

For z/OS 1.9 and above, KERBLVL is ignored, all levels are supported.

When used with the SDT, this keyword has the format:

```
TSS ADDTO(SDT) REALM(KERBDFLT|foreign_realm)
               REALMNAME(realmname)
               KERBPASS(password)
               CHKADDRS
               [ENCRYPT('[DES|NODES]
                        [DES3|NODES3]
                        [DESD|NODESD]
                        [AES128|NOAES128]
                        [AES256|NOAES256]')]
```

Note: The CHKADDRS keyword is used only with KERBDFLT realm record, not foreign realms.

When used with an ACID, this keyword has the format:

```
TSS ADDTO(acid) KERBNAME(kerbname)
                [ENCRYPT('[DES|NODES][DES3|NODES3][DESD|NODESD]
                         [AES128|NOAES128][AES256|NOAES256]')]
                [MAXTKTLF(ticket-life)]
```

Note: When ENCRYPT has been successfully added to an ACID, it appears in the LIST command as ENCTYPE.

If ENCRYPT is not specified, ENCRYPT('DES DES3 DESD AES128 AES256') is the default. Unless the negative keyword is explicitly specified, the positive keyword is assumed.

This keyword is used with:

- The commands ADDTO, REPLACE, and REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- MISC3(SDT) authority to ADDTO or REMOVE records or REPLACE fields in the SDT
- MISC8(LISTSDT) authority to list realms in the SDT
- ACID(MAINTAIN) and MISC4(KERBUSER) authority to update the KERBNAME of an ACID and its associated fields

EXCLUDE Keyword—Date Exclusion

Valid on z/OS, z/VSE, and z/VM.

Use the EXCLUDE keyword to add, remove, or replace a list of dates that are specifically excluded from the calendar record in the SDT Record.

This keyword has the following format:

```
TSS ADD(SDT) CALENDAR(cal-name)
             [YEAR(yyyy)]
             [DAYS(days)]
             [INCLUDE(mm/dd,...]
             [EXCLUDE(mm/dd,...)]
```

mm/dd

Lists specific dates (two digit month “mm”, two-digit day “dd”) that are excluded in addition to the ones added with the DAYS keyword, or INCLUDE keyword.

Note: The format of the date in EXCLUDE is unaffected by the DATE control option. EXCLUDE has no effect unless the date had previously included TSS Commands using DAYS or INCLUDE.

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The SDT Record exclusively
- MISC3(SDT) authority

Example: EXCLUDE keyword

This example creates a calendar for the present year with April 16 disabled:

```
TSS ADDTO(SDT) CALENDAR(CAL1)
               EXCLUDE(04/16)
               DAYS(WEEKDAY)
```

EXPIRES Keyword—Expiration Removal

Valid on z/OS, z/VM, and z/VSE.

Use the EXPIRES keyword with the REMOVE command function to remove an expiration that had been set using the FOR or UNTIL parameters

This keyword has the following format:

```
TSS REMOVE(acid) EXPIRES
```

This keyword is used with:

- The REMOVE command
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Example: EXPIRES keyword

This example provides access to a temporary AUDIT01 ACID:

```
TSS REMOVE(AUDIT01) EXPIRES
```

FACILITY Keyword—Specify Facilities

Valid on z/OS, z/VM, and z/VSE.

Use the FACILITY keyword to:

- Specify which facility or facilities an ACID may or may not access
- Give or remove administrators authority to administer the use of a facility (z/VM and z/VSE) or facilities (in z/OS) within their scope
- Permit an ACID to have access to a resource through the specified facility

When used with ADMIN, this keyword has the following format:

```
TSS ADMIN(acid) FACILITY(facility name(s))
```

When used with ADDTO, this keyword has the following format:

```
TSS ADDTO(ACID|ALL) FACILITY(fac,fac,.... | ALL)
```

When used with PERMIT, this keyword has the following format:

```
TSS PERMIT(acid) resource(prefix(es))
                 FACILITY(facility name)
```

If the FACILITY keyword is not specified in a PERMIT function, CA Top Secret allows the user to access the resource from any facility that is authorized for that user.

Note: Specifying the FACILITY keyword on a PERMIT command does *not* require the administrator to have authority to grant an ACID signon ability to the facility. Therefore, an administrator may enter this command even though the administrator does not have authority for the facility name. However, the administrator must have XAUTH authority for the resource class.

Facilities listed as operands must reside in the site's Facilities Matrix Table. This is done by specifying the ACTIVE sub-option of the FACILITY control option for that facility.

The control options that can be used with FACILITY are:

Expiration

FOR or UNTIL

Time/Day

TIME and/or DAY; TIMEREC and/or CALENDAR

Actions

FAIL, NOTIFY, AUDIT, DENY

System access

SYSID

Note: You can also control mode by facility for an individual user by using the FACILITY keyword on a TSS PERMIT command. For example:

```
TSS PERMIT(USER01) MODE(WARN)
                   FACILITY(BATCH,TSO)
```

This keyword is used with:

- The commands CREATE, PERMIT, ADMIN, DEADMIN, ADDTO, REPLACE, REMOVE, REVOKE, and WHOHAS
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ALL Record
- Authorization to administer a specific facility or facilities to authorize access to ACIDs

Examples: FACILITY keyword

This example authorizes user NJ014 to access both z/VM and TSO:

```
TSS ADDTO(NJ014) FACILITY(VM,TSO)
```

This example authorizes USER05 to sign on to CICS on Monday, Wednesday, and Friday from the hours of 9 a.m. to 5 p.m. for the next 10 days and to audit the ACID's activity:

```
TSS ADDTO(USER05) FACILITY(CICS)
                  DAYS(MON,WED,FRI)
                  TIMES(9,17)
                  FOR(10)
                  ACTION(AUDIT)
```

This example authorizes a user to access all facilities currently active in the installation:

```
TSS ADDTO(acid) FACILITY(ALL)
```

Note: This authorization also gives the user access to all subsequently added facilities.

This example removes access to a facility:

```
TSS REMOVE(NJ014) FACILITY(TSO)
```

Note: If NJ014 had been authorized to access all facilities via TSS ADDTO(acid) FACILITY(ALL), this command would not remove access to the TSO facility. Instead use ACTION(DENY).

This example authorizes a user to access all facilities except CICSPROD:

```
TSS ADDTO(ACID) FACILITY(ALL)

TSS ADDTO(ACID) FACILITY(CICSPROD)
                ACTION(DENY)
```

This example allows all users to access the TSO facility between the hours of 9 a.m. and 5 p.m:

```
TSS ADDTO(ALL) FACILITY(TSO)TIME(09,17)
```

This example allows the user to use the TSO facility, but only on the system that has an SMFID of TSO1:

```
TSS ADD(useracid) FACILITY(TSO) SYSID(TS01)
```

This example allows user ED001 to have READ access (default) to tutorial data sets through both TSO and BATCH:

```
TSS PERMIT(ED001) DSNC(TUTORIAL.DATASET)
                  ACC(READ)
                  FAC(TSO,BATCH)
```

In this example, an SCA gives an administrator the authority to create and maintain ACIDs for CICS users:

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

This entry allows the TSS administrator to create ACIDs for CICS users or profiles, and would allow the administrator to use various TSS functions (for example ADDTO, MOVE, and RENAME) to maintain those ACIDs.

CA Top Secret does not check an administrator's facility authority when the administrator specifies a FACILITY within a TSS PERMIT function.

Note: Specifying ALL gives the security administrator the authority to add any facility to an ACID.

FAILLOAD Keyword—SIGVER record program definition

Valid on z/OS.

Use the FAILLOAD keyword to specify what would trigger z/OS to fail a program load if an error is encountered during signature verification.

```
TSS ADD(SIGVER) PROGRAM(PROG01) FAILLOAD(ANYBAD|BADSIGNONLY|NEVER)
```

Valid values

ANYBAD

Fails the load on any error.

BADSIGNONLY

Fails the load only if the signature is bad.

NEVER

Never fails the load

Capacity of list

One keyword per SIGVER program entry.

This keyword is used with:

- The commands ADDTO and REPLACE
- The SIGVER record only
- MISC5(SGVADMIN) authority to maintain the SIGVER record.

Examples: FAILLOAD keyword

This example adds program PROG01 with ANYBAD attribute of FAILLOAD keyword:

```
TSS ADDTO(SIGVER) PROGRAM(prog01) FAILLOAD(anybad)
```

FDTCODE Keyword—Field Code Data

Valid on z/OS, z/VM, and z/VSE.

Use the FDTCODE keyword to:

- Allow the CA Top Secret administrator to list data from the FDT Record
- Add user-defined field names to the FDT Record

When used with LIST, this keyword has the following format:

```
TSS LIST(FDT) FDTCODE(code)
```

When used with ADDTO, this keyword has the following format:

```
TSS ADDTO(FDT) FDTNAME(field name)
               FDTCODE(hex code)
               MAXLEN(nnnnn)
```

hex code

A unique two-digit hexadecimal code.

Range: 01 to FF

FDTCODE (as well as MAXLEN and FDTNAME) is required when adding a resource to the FDT Record.

This keyword is used with:

- The commands ADDTO and LIST
- The FDT ACID only
- MISC1(RDT) authority to ADDTO field names in the FDT Record
- MISC1(RDT) or MISC8(LISTRDT) authority to LIST the FDT Record

Example: FDTCODE keyword

This example lists data concerning how field code 3F is processed:

```
TSS LIST(FDT) FDTCODE(3F)
```

This example adds PERSDEPT to the FDT Record:

```
TSS ADDTO(FDT) FDTNAME($PERSDEPT)
               FDTCODE(3F)
               MAXLEN(40)
```

This example removes $PERSDEPT from the FDT Record:

```
TSS REMOVE(FDT) FDTNAME($PERSDEPT)
```

FDTNAME Keyword—FDT Record Field Names

Valid on z/OS, z/VM, and z/VSE.

Use the FDTNAME keyword to:

- List data from the FDT Record about how the specified field is processed
- Add or remove user-defined field names to or from the FDT Record

When used with LIST, this keyword has the following format:

```
TSS LIST(FDT) FDTNAME(fieldname)
```

When used with ADDTO, this keyword has the following format:

```
TSS ADDTO(FDT) FDTNAME(fieldname)
               FDTCODE(hex code)
               MAXLEN(nnnnn)
```

fieldname

Specifies the field.

Length: 1 to 8 characters.

Capacity: One resource class per command

Notes:

- FDTNAME, FDTCODE, and MAXLEN are required when adding a resource to the FDT Record.
- To prevent any possibility of your field name conflicting with any future CA Top Secret resources, include either a numeric (0 to 9) or a national ($ # @) in the FDTNAME name.
- The first four characters of all field names must be unique and cannot conflict with any dynamically-defined or predefined field name.
- Fields can be removed from the FDT even if a user ACID contains those fields. If a new FDT entry is defined using the same FDTCODE, the results are unpredictable. A TSS LIST of a user ACID removes any field information no longer in the FDT.

This keyword is used with:

- The commands REPLACE, LIST, ADDTO, and REMOVE
- The FDT only
- MISC1(RDT) authority to ADDTO or REMOVE field names in the FDT Record
- MISC1(RDT) or MISC8(LISTRDT) authority to LIST the FDT Record

Examples: FDTNAME Keyword

This example adds $PERSDEPT to the FDT Record:

```
TSS ADDTO(FDT) FDTNAME($PERSDEPT)
               FDTCODE(3F)
               MAXLEN(11)
```

This example removes $PERSDEPT from the FDT Record:

```
TSS REMOVE(FDT) FDTNAME($PERSDEPT)
```

FIRST Keyword—Add a Profile to the Beginning of a List

Valid on z/OS and z/VM.

Use the FIRST keyword to enable an administrator to add a profile to the beginning of a profile list.

This keyword has the following format:

```
TSS ADDTO(acid) PROFILE(new profile)
                FIRST
```

This keyword is used with:

- The ADDTO command
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Example: FIRST keyword

This example adds the profile FINPROF1 as the first one in USER01's profile list:

```
TSS ADDTO(USER01) PROFILE(FINPROF1)
                  FIRST
```

FOR Keyword—Specify ACID's or Permission's Life

Valid on z/OS, z/VM, and z/VSE.

Use the FOR keyword to:

- Specify the number of days that an ACID may be used before it expires
- With SUSPEND to specify a period of time during which an ACID is suspended
- Specify the number of days that the ACID is permitted to access the resource

This keyword has the following format:

```
TSS PERMIT|ADDTO(acid) FOR(nnn)
```

nnn

The number of days until the ACID can be used for. Use 0 for no expiration.

Range: 0 to 255

This keyword is used with:

- The commands CREATE, PERMIT, REPLACE, ADDTO, REVOKE, and REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Note: When used with SUSPEND, administrators must also have MISC1(SUSPEND) authority.

FOR Related Keywords

Note the following related keywords:

- FOR and UNTIL are mutually exclusive.
- Use the SUSPEND keyword with FOR to specify a time period during which an ACID is suspended.

- To REMOVE the expiration interval use:
  `TSS REMOVE(acid) EXPIRE`
- You cannot REMOVE the FOR attribute from a facility, remove the facility from the ACID:
  `TSS REMOVE(acid) FACILITY(facility)`

Expired Access

When a resource permission expires, the permission is ignored by the CA Top Secret search algorithm. The TSS LIST function automatically removes the expired resource permission.

Examples: FOR keyword

This example forces an ACID to expire after 5 days:

```
TSS ADDTO(USER23) FOR(5)
```

This example totally removes the expiration limit from USER23 and thus reactivate the ACID:

```
TSS REMOVE(USER23) FOR(5)
```

This example removes the expiration:

```
TSS ADDTO(USER23) FOR(0)
```

If the FOR keyword is ADDed to an ACID in combination with the SUSPEND keyword, the ACID is temporarily suspended for the specified number of days and can then be used again. This example suspends USER23 for 14 days:

```
TSS ADDTO(USER23) SUSPEND
                  FOR(14)
```

This example uses the FOR keyword to add PROFILE ACIDs and FACILITY(s) to a user:

```
TSS ADDTO(USER23) FACILITY(TSO)
                  FOR(5)
```

Note: When used with a PROFILE or FACILITY, the use of that profile or facility expires, not the ACID itself.

This example indicates that USER01 is allowed READ access (default) to any data set suffixed by .FILE for one day (for the rest of today):

```
TSS PERMIT(USER01) DSNAME(***.FILE)
                   FOR(1)
```

This example revokes USER01's access:

```
TSS REVOKE(USER01) DSNAME(***.FILE)
```

FORCE Keyword—Bypass Checks

Valid on z/OS.

Use the FORCE keyword to perform either of the following activities:

- Perform a rollover unconditionally (replacing an old certificate with a new certificate).
- Remove a certificate that has been the target of a GENREQ command from a user.

When you specify the FORCE keyword, the following checks are not performed:

- The values of DIGICERT and NEWDIGIC must be different.
- The certificates identified by DIGICERT and NEWDIGIC must both have private keys associated with them.
- The certificate identified by NEWDIGIC must have never been the target of a previously issued ROLLOVER command and never used to sign other certificates.
- The original certificate must not have had a GENREQ issued against it.

Important! If DIGICERT and NEWDIGIC have the same value, and you have also specified the FORCE keyword, the product deletes the private key of the certificate.

This keyword has the following format:

```
TSS ROLLOVER(acid) DIGICERT(old_certificate_id)
                   NEWDIGIC(new_certificate_id)
                   FORCE
```

acid

Specifies the user ACID associated with the certificate.

This keyword is used with:

- The REMOVE and ROLLOVER commands
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority for users
- MISC4(CERTSITE) authority for CERTSITE ACIDs
- MISC4(CERTAUTH) authority for CERTAUTH ACIDs

Example: Replace the TEN Certificate with the NINE Certificate

This example completes the re-keying of the TEN certificate by replacing the TEN certificate with the NINE certificate:

```
TSS ROLLOVER(CERTSITE) DIGICERT(NINE)
                       NEWDIGIC(TEN)
                       FORCE
```

FORMAT Keyword—Specify Certificate Format

Valid on z/OS.

Use the FORMAT keyword to specify the output format when exporting a digital certificate to an output data set.

This keyword has the following format:

```
TSS EXPORT(acid) DIGICERT(name)
                 DCDSN(dsname)
                 FORMAT(format-type)
                 PKCSPASS(PK12 password)
```

CERTB64

(Default) Indicates Base64 encoded certificates.

CERTDER

Indicates DER encoded X.509 certificates.

PKCS12B64

Indicates DER encoded (then Base64 encoded) PKCS#12 package.

PKCS12DER

Indicates DER encoded PKCS#12 package.

PKCS7B64

Specifies a B64 encoded PKCS#7 package.

PKCS7DER

Specifies a DER encoded PKCS#7 package.

If the certificate's private key resides in an ICSF storage facility and the format of PKCS12DER or PKCS12B64 is specified in the TSS EXPORT command, the command is rejected and a TSS0533E message is issued. You cannot export a digital certificate with ICSF.

ICSF is the interface to the cryptographic hardware on z/OS. You must have cryptographic hardware installed and enabled on your system.

This keyword is used with:

- The commands CHECKCERT and EXPORT
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority for users
- MISC4(CERTEXPO) and MISC4(CERTSITE) authority for CERTSITE ACIDs
- MISC4(CERTAUTH) authority for CERTAUTH ACIDs

Examples: FORMAT keyword

This example exports a digital certificate to a data set in Base64 format:

```
TSS EXPORT(user1) DIGICERT(cert0001)
                  DCDSN(udrt1.cert.data)
                  FORMAT(certb64)
                  PKCSPASS(secreto
```

This example exports a digital certificate to a data set in DER encoded X.509 format:

```
TSS EXPORT(user1) DIGICERT(cert0001)
                  DCDSN(udrt1.cert.data)
                  FORMAT(certder)
                  PKCSPASS(secreto
```

This example exports a digital certificate to a data set in Base64 PKCS#12 package format:

```
TSS EXPORT(user1) DIGICERT(cert0001)
                  DCDSN(udrt1.cert.data)
                  FORMAT(PKCS12B64)
                  PKCSPASS(secretone)
```

This example exports a digital certificate to a data set in DER encoded PKCS#12 package format:

```
TSS EXPORT(user1) DIGICER(cert0001)
                  DCDSN(udrt1.cert.data)
                  FORMAT(PKCS12DER)
                  PKCSPASS(secreto
```

FROMICSF Keyword—Specify Certificate Label

Valid on z/OS.

Use the FROMICSF keyword to specify an optional label associated with the certificate private or public key being stored into the ICSF storage facility. Up to 64 characters can be specified for the label name.

This label is used as a descriptive identifier for a certificate ICSF key, and must be unique across the mvs complex.

This keyword has the following format:

```
TSS GENCERT(acid) FROMICSF( PKDS–label–name/*)
```

PKDS-label-name

(Optional) Specifies that the public key for this certificate will be obtained from ICSF using the specified PKDS label. The private key of the source certificate, if one exists, will not be associated with the new certificate. FROMICSF cannot be specified with the DCDSN or LABLPKDS parameters. SIGNWITH must be specified when FROMICSF is specified.

The PKDS label must conform to ICSF label syntax rules. The first character must be alphabetic or national. The field is folded to uppercase.

Valid characters: Alphanumeric, national (@,#,$) or period(.).

Range: Up to 64 characters

This keyword is used with:

- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- The commands GENCERT
- ACID(MAINTAIN) for users within their scope, and MISC4(CERTGEN)
- MISC4(CERTSITE) for SITE
- ACID and MISC4(CERTAUTH) for CERTAUTH ACID.

Examples: FROMICSF keyword

This example specifies the optional FROMICSF label:

```
TSS GENCERT(user1) DIGICERT(cert0001) SIGNWITH(certauth,auth0001)
FROMICSF(userx.certicsf)
```

GAP Keyword—Specify if a Profile is Globally Administered

Valid on z/OS and z/VM.

Use the GAP keyword to specify that a profile is, or will cease to be, globally administered.

Only an administrator with the same scope of authority as the administrator who CREATED the profile can assign the GAP attribute to that profile.

This keyword has the following format:

```
TSS ADDTO(profile acid) GAP
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE.
- The ACID type profile

Examples: GAP keyword

This example creates a profile ADMACT:

```
TSS CREATE(ADMACT) TYPE(PROFILE)
                   NAME('ACTPROJECT')
                   DEPARTMENT(ACCTD)

TSS PERMIT(ADMACT) DSNAME(*.ACCTS)
                   ACCESS(READ)
```

Since many users in several departments required access to profile ADMACT, the SCA decided to ADDTO the GAP attribute to the profile, allowing CA Top Secret administrators in all departments to administer the profile:

```
TSS ADDTO(ADMACT) GAP
```

This example removes Global Administration from a profile when no longer required:

```
TSS REMOVE(ADMACT) GAP
```

GID Keyword—Assign a Group ID for USS Group Security

Valid on z/OS and z/VM.

Use the GID keyword to assign a numeric group ID (GID) value to a group ACID for security within USS.

Note: You need administrative privileges to use this keyword.

The assigned GID is shared by one or more ACIDs by adding the GROUP to every ACID that is allowed to use it. GIDs can be explicitly assigned to a GROUP or automatically assigned within a numeric range by CA Top Secret. You can explicitly specify a range for the assignment, or you can accept the range default in the DFLTRNGG control option.

Important! After you add a GID to a group ACID, specify TSS MODIFY(OMVSTABS) to perform a refresh and allow the assignment to take effect. Attempting to use the GID assignment before an OMVSTABS refresh could interfere with the ACID's attempts to use USS services.

This keyword has the following format:

```
TSS ADDTO(acid) GID(uss_group_id)

TSS ADDTO(acid) GID(?)

                [RANGE(low_gid,high_gid)]
```

acid

Specifies the group ACID to which you are assigning a USS GID.

uss_group_id

Specifies a numeric GID to assign to the ACID. The GID is uniquely assigned to an ACID of type GROUP.

Range: 1 to 2147483647

?

Instructs CA Top Secret to automatically assign the USS GID. If you specify RANGE in the command, the product assigns the first unused positive integer that is greater than or equal to the *low_gid* value and less than or equal to the *high_gid* value. If you do not specify RANGE, the product assigns the first available integer in the range defined by DFLTRNGG control option.

Note: You *cannot* use the ? value with REMOVE.

If you are using the Command Propagation Facility (CPF) and CPFAUTOGID(YES), the TSS command is transmitted with the assigned GID value instead of the ? value (for example, TSS ADD(JONATHAN) GID(256) instead of TSS ADD(JONATHAN) GID(?)).

RANGE

Specifies a range from which the product will choose an available GID (if you have also specified ? in the command syntax to automatically assign a GID).

low_gid

Indicates the lowest assignable GID available to this ACID.

Range: 1 to 2147483647

high_gid

Indicates the highest assignable GID available to this ACID.

Range: 1 to 2147483647

This keyword is used with the commands ADDTO, REMOVE, REPLACE, WHOHAS, and WHOOWNS.

Example: Assign a GID to a Group ACID

This example assigns GID 93456 to GRPACID1:

```
TSS ADDTO(GRPACID1) GID(93456)
```

Note: After assigning the GID to the ACID, issue TSS MODIFY(OMVSTABS) for the assignment to take effect.

Example: Remove a GID Assignment from an ACID

This example removes the connection between a GID and an ACID:

```
TSS REMOVE(GPACID1) GID(93456)
```

Example: Automatically Assign a GID Through the DFLTRNGG Control Option

This example automatically assigns a number to an ACID (using an available integer in the default range defined by the DFLTRNGG control option):

```
TSS ADDTO(GPACID1) GID(?)
```

Example: Automatically Assign a GID By Using an Available Integer in a Range That You Specify

This example automatically assigns a number to an ACID in the inclusive range 10,000 to 200,000:

```
TSS ADDTO(GPACID1) GID(?)
                   RANGE(10000,200000)
```

GROUP Keyword—Define Groups to an ACID

Valid on z/OS, z/VM, and z/VSE.

Use the GROUP keyword to add or remove groups from a specified ACID. Up to five GROUP ACIDs can be added per TSS command.

Note: The GROUP and PROFILE keywords:

- Cannot be used in the same TSS commands
- Are interchangeable and coding GROUP() removes both GROUPS and PROFILE

This keyword has the following format:

```
TSS ADDTO(acid) GROUP(group acid,...)
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Examples: GROUP keyword

This example indicates that group WRIT01 is now connected to USER02:

```
TSS ADDTO(USER02) GROUP(WRIT01)
```

This example removes the connection between a group and an ACID:

```
TSS REMOVE(USER02) GROUP(WRIT01)
```

This example uses the FOR keyword to connect GROUP ACIDs to a user.

```
TSS ADDTO(USER02) GROUP(WRIT01)
                  FOR(5)
```

When used with a GROUP, the use of that group expires, not the ACID itself.

This example removes the GROUP from the ACID (You cannot remove the FOR attribute from a GROUP):

```
TSS REMOVE(USER02) GROUP(WRIT01)
```

HOME Keyword (MODLUSER)—Define a Prototype HOME Subdirectory

Valid on z/OS and z/VM.

Use the HOME keyword to add a prototype HOME directory to an ACID specified by the MODLUSER control option. The specified directory is used for users that do not have an OMVS segment. When a session user attempts to use UNIX Systems Services (USS) but does not have USS credentials, the UNIQUSER and MODLUSER control options dynamically provide the credentials in the user's ACID. The HOME keyword in a MODLUSER ACID provides a fixed or variable directory name that is copied into the HOME keyword of the session ACID.

This keyword has the following format:

```
TSS ADDTO(acid) HOME(/sub1/sub2/..../subn)
```

acid

Specifies the ACID that you identified with the MODLUSER control option. This ACID contains a model for a default OMVS segment.

/sub1/sub2/…./subn

Specifies one or more valid USS directories. The first character of each specified directory must be a forward-slash (/). Successive subdirectories in this subdirectory must each begin with a forward-slash(/).

Range: 1 to 1024 characters

Note: For information about allowable characters, see the IBM *Unix Systems Services Command Reference*.

You can use a variable (&acid) to represent a subdirectory. When the USS credentials are copied from the MODLUSER to the OMVS segment of the session ACID, this variable is replaced by the session ACID. When you use this variable, the following rules apply:

- The variable should be used only as a MODLUSER prototype. Do not use the variable in a session ACID.
- Using the variable as the final subdirectory in the HOME directory string makes the value compatible with the USS "automount" generic substitution. If you use the variable in a position other than the final specified subdirectory, or you do not use automount, ensure that the HOME directory generated for the session ACID has been created with appropriate mkdir command and has been primed with appropriate security flags or HFSSEC permissions.

  Note: For information about automount, see the z/OS release *UNIX Systems Services Planning Guide.*
- Specifying &ACID substitutes an uppercase version of the session ACID. Specifying &acid substitutes a lowercase version of the session ACID.

This keyword is used with:

- The commands ADDTO, REMOVE, REPLACE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Example: Use a Variable When Defining a Prototype Subdirectory to a MODLUSER ACID

This example indicates that the prototype subdirectory /u/a1/&acid is now connected to MODL02:

```
TSS ADDTO(MODL02) HOME(/u/a1/&acid)
```

CA Top Secret will copy OMVS credentials from MODL02 in the following situation:

- UNIQUSER(YES) and MODLUSER(MODL02) are set as control options.
- USER05 has no USS credentials but attempts to perform USS services.

This subdirectory contains the lowercase symbolic variable; therefore, the new HOME for USER05 will be /u/a1/user05.

Example: Define a Specific Prototype Subdirectory to a MODLUSER ACID

This example connects a specific uppercase prototype to MODLUSER ACID MODL02:

```
TSS ADD(MODL02) HOME(/LIFE/OF/PI)
```

CA Top Secret will copy OMVS credentials from MODL02 in the following situation:

- UNIQUSER(YES) and MODLUSER(MODL02) are set as control options.
- USER05 has no USS credentials but attempts to perform USS services.

The new HOME for USER05 will be /LIFE/OF/PI.

Example: Remove a HOME Directory Specification from a MODLUSER

To remove a HOME directory specification from a MODLUSER, you may specify HOME with or without an operand. This example shows syntax for both methods:

```
TSS REMOVE(MODL02) HOME(/LIFE/OF/PI)
TSS REMOVE(MODL02) HOME
```

HOME Keyword (Session User)—Define USS Entry Subdirectory

Valid on z/OS and z/VM.

Use the keyword to specify a valid HOME directory for entry into UNIX Systems Services (USS).

A session or batch job that enters a USS shell or uses a USS callable service performs an implicit change-directory command to the HOME directory. Subsequent processing depends on the value of the HFSSEC control option:

- If HFSSEC(OFF) is set, the security bits of the directory are checked against the security bits of the session user.
- If HFSSEC(ON) is set, the HOME directory permissions established for the user (and associated profiles) are validated.

Violations cause the session to terminate or cause the service to return an error.

This keyword has the following format:

```
TSS ADDTO(acid) HOME(/sub1/sub2/…./subn)
```

acid

Specifies the ACID to which you are defining the USS entry subdirectory.

/sub1/sub2/…./subn

Specifies a valid USS directory. The first character must always be forward-slash (/). Successive subdirectories in this subdirectory must begin with a forward-slash (/).

Range: 1 to 1024 characters

Note: For information about allowable characters, see the IBM *Unix Systems Services Command Reference*.

This keyword is used with:

- The commands ADDTO, REMOVE, REPLACE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Example: Define a USS Subdirectory for an ACID

This example indicates that subdirectory ABC0021 is now connected to USER02:

```
TSS ADDTO(USER02) HOME(/ABC0021)
```

Example: Remove a Subdirectory Definition from an ACID

This example removes the connection between a subdirectory and an ACID:

```
TSS REMOVE(USER02) HOME(/ABC0021)
```

ICSF Keyword—Place Private Key in ICSF

Valid on z/OS.

Use the ICSF keyword to indicate that the generated private key is placed in ICSF. If the DSN parameter was also specified and an existing certificate is replaced, the certificate is also placed in ICSF. ICSF must be active and configured for PKA operations.

When specifying the ICSF parameter, the KEYSIZE parameter can be as high as 1024.

Consider the following:

- ICSF specified with ADD is ignored if no private key is involved
- If ICSF, PCICC, or LABLPKDS is not specified with ADD the key is stored in the security file as a non-ICSF key
- If ICSF is specified with ADD, but ICSF is not configured for PKA operations, the key is stored in the security file as a non-ICSF key
- If the key is stored in ICSF, the security file stores a label (which refers to the key)
- ICSF is only valid to create only RSA keys with a length of up to 1024 bytes
- If the certificate's private key resides in an ICSF storage facility and the format of PKCS12DER or PKCS12B64 is specified in the TSS EXPORT command, the command is rejected

This keyword has the following format:

```
TSS GENCERT(acid) DIGICERT(8–byte name)
                  ICSF

TSS REKEY(acid) DIGICERT(8-byte name)
                DCDSN(dsname)
                ICSF
```

This keyword is used with:

- The commands ADDTO, GENCERT, and REKEY and RENEW
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority for users
- MISC4(CERTUSER) authority for user ACIDs
- MISC4(CERTSITE) authority for CERTSITE ACIDs
- MISC4(CERTAUTH) authority for CERTAUTH ACIDs

Examples: ICSF keyword

This example indicates that the digital certificate is stored in an ICSF facility:

```
TSS ADDTO(USERA) DIGICERT(USERACER)
                 DCDSN(USERA.CERT01)
                 ICSF
```

This example removes the entry in the ICSF storage facility by removing the DIGICERT from the acid:

```
TSS REMOVE(USERA) DIGICERT(USERACER)
```

IDLABEL Keyword—Specify Identity Propagation Label

Valid on z/OS.

Use the IDLABEL keyword to specify the label associated with the Identity Propagation name filter.

This keyword has the following format:

```
TSS ADDTO(acid) IDMAP(recid) IDMAPDN('distributed-identify-username-filter')
IDMAPRN('DISTRIBUTED-IDENTIFY-REGIStryname') [IDLABEL('32-byte label')]
```

The label name must be enclosed in single quotation marks if embedded blanks are used.

Up to 32 characters can be specified for the label name. It can contain embedded blanks and mixed-case characters, and it is stripped of leading and trailing blanks. If a single quotation mark is intended to be part of the label name, use two single quotation marks together for each single quotation mark within the string, and the entire string must be enclosed within single quotation marks.

IDLABEL is optional on the TSS ADDTO IDMAP command. If it not included on the command, the record identifier name entered for IDMAP is automatically used for IDLABEL.

This keyword is used with:

- The commands ADDTO, LIST, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Example: IDLABEL keyword

This example shows IDMAP, IDMAPDN, IDMAPRN, and IDLABEL keyword usage in one TSS ADDTO command.

The following command adds the JDoe ACID and specifies the IDMAP, IDMAPDN, IDMAPRN, and IDLABEL:

```
TSS ADD(JDoe) IDMAP(JDoeIDM1) IDMAPDN('JDoe Distinguished name') IDMAPRN('JDoe Registry name') IDLABEL('JDoe ID Propagation Record')
```

The following command lists all fields associated with the specified IDLABEL:

```
TSS LIST(JDoe) IDLABEL('JDoe ID Propagation Record')
```

IDMAPDN, IDLABEL, and IDMAPRN Usage Rules

When using these keywords, note the following rules:

- The combination of IDMAPDN and IDMAPRN must be unique across all IDMAP records. If it is not, an error message that shows the existing record ID will appear.
- The combination of Recid (Logonid portion) and IDLABEL must be unique across all IDMAP records. If it is not, an error message that shows the existing record ID will appear.

IDMAP Keyword—z/OS Identity Propagation Mapping

Valid on z/OS.

Uses the IDMAP keyword to map a distributed identify username and a distributed registry name to a CA Top Secret ACID.

Note: IDMAP (Identity Propagation) requires VSAM II setup.

IDMAP specifies a unique eight-byte record identifier. When the Identity Propagation name filter is defined, the Identity Propagation mapping information is stored in an IDMAP record in the VSAM file.

This keyword has the following format:

```
TSS ADDTO(acid) IDMAP(recid) IDMAPDN('distributed-identify-username-filter')
IDMAPRN('DISTRIBUTED-IDENTIFY-REGIStryname') [IDLABEL('32-byte label')]
```

This keyword is used:

- The commands ADDTO, LIST, REMOVE, REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

More Information:

Activating IDMAP Profile Records (see page 230)

Example: IDMAP Keyword

The following command lists all fields (ACID, distinguished name, label, registry name, and IDMAP) associated with the specified IDMAP:

```
TSS LIST(JDoe) IDMAP(JDoeIDM1)
```

The following command lists all IDMAP information:

```
TSS LIST(JDoe) IDMAP(ALL)
```

Activating IDMAP Profile Records

When IDMAP records are inserted, changed, or deleted, CA Top Secret updates its in-core table of IDMAP records. The in-core table is used when mapping a distributed user to a Logonid. The in-core table is also used when checking for the uniqueness of the IDMAPDN and IDMAPRN combination and the Recid (Logonid portion) and IDLABEL combination.

Therefore, any changes to the IDMAP records become effective immediately. Old records are maintained in the table in a *no longer used* state. If many changes have been made, clean up the table by issuing the F TSS,REFRESH(IDMAPTAB) console command

If changes to IDMAP records are made on one system in a CPF environment, the in-core tables on the other systems will not be updated automatically. In this case, issue the F TSS,REFRESH(IDMAPTAB) console command on the other systems.

IDMAPDN Keyword—Specify Identity Distinguished Name

Valid on z/OS.

IDMAPDN can be specified as the full Distinguished Name, the Distinguished Name filter (when it is in x.500 format) or an asterisk (*). The IDMAPDN is the Distinguished Name. IDMAPDN is stored in UTF-8 format. IDMAPDN is a required field. It can be up to 246 bytes long in UTF-8 format.

An IDMAPDN may contain any of the following special characters:

Backslash \	Comma ,	Equal sign =
Greater than >	Less than <	Plus sign +
Double quote "	Semicolon ;	

These special characters must be escaped by a backslash (\) unless they are an equal sign (=) or a comma (,), or a semicolon (;) that is meant to be a delimiter in an x.500 Distinguished Name. Semicolons are changed to and treated as commas. The attribute type is forced to uppercase. The attribute value can be mixed case. For example, if the Distinguished Name is as sample A, then you must specify it as shown in Sample B:

Sample A

```
UID=Jane<Smith>Doe,OU=XYZ+Flattener,O=XYZ Company,C=USA
```

Sample B

```
UID=Jane\<Smith\>Doe,OU=XYZ\+Flattener,O=XYZ Company,C=USA
```

When the distinguished name is in the x.500 format, the order of the RDNs (the sections separated by commas) needs to be in order from most specific to least specific.

This keyword has the following format:

```
TSS ADDTO(acid) IDMAP(recid) IDMAPDN('distributed-identify-username-filter')
IDMAPRN('DISTRIBUTED-IDENTIFY-REGIStryname') [IDLABEL('32-byte label')]
```

This keyword is used with:

- The commands ADDTO, LIST, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

More Information:

IDMAPDN, IDLABEL, and IDMAPRN Usage Rules (see page 229)
Activating IDMAP Profile Records (see page 230)
Using Filters for the Distinguished Name and Registry Name (see page 232)

Example: IDMAPDN keyword

The following command replaces the distinguished name:

```
TSS REP(JDoe) IDMAP(JDoeIDM1) IDMAPDN('JDoe NEW Distinguished name')
```

Using Filters for the Distinguished Name and Registry Name

IDMAPDN can be specified as the full Distinguished Name, the Distinguished Name filter (when it is in x.500 format), or an asterisk (*).

IDMAPRN can be specified as the full Registry Name or an asterisk (*).

An asterisk (*) for the Distinguished Name or for the Registry Name indicates that it will match anything.

When the Distinguished name is in x.500 format, it must be specified in the order from most specific Relative Distinguished name (RDN) to least specific RDN. An RDN is a section of the Distinguished Name, and they are separated by commas.

For example, suppose the x.500 format Distinguished Name is:
UID=JohnDoe,OU=User,O=XYZ

UID=JohnDoe is the most specific RDN. O=XYZ is the least specific RDN.

For this example, the search order would be as folllows:

1. UID=JohnDoe,OU=User,O=XYZ + Full Registry Name
2. UID=JohnDoe,OU=User,O=XYZ + '*'
3. OU=User,O=XYZ + Full Registry Name
4. OU=User,O=XYZ + '*'
5. O=XYZ + Full Registry Name
6. O=XYZ + '*'
7. '*' + Full Registry Name
8. '*' + '*'

When the Distinguished Name is not in the x.500 format, the search order would be as follows:

1. Full Distinguished Name + Full Registry Name
2. Full Distinguished Name + '*'
3. '*' + Full Registry Name
4. '*' + '*'

IDMAPRN Keyword—Specify Identity Registry Name

Valid on z/OS.

IDMAPRN can be specified as the full Registry Name or an asterisk (*). The IDMAPRN is the Registry Name in UTF8 format. IDMAPRN is stored in UTF-8 format. IDMAPRN is a required field. It can be up to 255 bytes long in UTF-8 format.

This keyword has the following format:

```
TSS ADDTO(acid) IDMAP(recid) IDMAPDN('distributed-identify-username-filter')
IDMAPRN('DISTRIBUTED-IDENTIFY-REGIStryname') [IDLABEL('32-byte label')]
```

This keyword is used with:

- The commands ADDTO, LIST, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Note:

The combination of IDMAPDN and IDMAPRN must be unique across all IDMAP records. If it is not, an error message appears.

The combination of USERID and IDLABEL must be unique across all IDMAP records. If it is not, an error message appears.

More Information:

IDMAPDN, IDLABEL, and IDMAPRN Usage Rules (see page 229)
Activating IDMAP Profile Records (see page 230)
Using Filters for the Distinguished Name and Registry Name (see page 232)

IDNFILTR Keyword—Specify a Distinguished Name Filter

Valid on z/OS.

Use the IDNFILTR keyword to specify the significant portion of the issuer's distinguished name used as a filter when associating an ACID with a digital certificate. The IDNFILTR data must match a portion of the issuer's distinguished name extracted from the certificate. The distinguished name from the point of the match to the end of the name is used as the filter data.

When specified with DCDSN, the filter must correspond to a starting point within the issuer's distinguished name found in the certificate contained in the data set. Specify enough of the name to precisely identify the starting point for the filter. For example, the certificate in the data set has an issuer as shown below and you want all certificates issued by BobsCertAuth selected by this filter:

```
OU=Class 1 Cerificate.O=BobsCertAuth,Inc.L=internet.C=US
```

Specify:

```
IDNFILTR('O=BobsCertAuth')
```

Without the data set containing the certificate, enter the following to produce the same result:

```
IDNFILTR('O=BobsCertAuth,Inc.L=internet.C=US')
```

IDNFILTR is optional if the subject's distinguished name filter (SDNFILTR) is specified. If IDNFILTR is not specified, only the subject's name is used as the filter. If IDNFILTR is specified and only a portion of the issuer's name is used as the filter, SDNFILTR must not be specified. If both IDNFILTR and SDNFILTR are specified, the IDNFILTR value does not need to begin with a valid prefix. This allows the use of certificates from a certificate authority that chooses to include non-standard data in the issuer's distinguished name.

A maximum of 255 characters can be entered for IDNFILTR if SDNSIZE(255) control option is specified. A maximum of 1024 characters can be entered for IDNFILTR if SDNSIZE(1024) control option is specified. When a starting value is specified for a certificate contained in a data set, there cannot be more than 255/1024 characters between the starting point and the end of the issuer's name in the certificate.

This keyword has the following format:

The IDNFILTR value must be enclosed in quotes.

```
TSS ADDTO(acid) CERTMAP(recid)
                IDNFILTR('issuer-dist-name-filter')
```

'*issuer-dist-name-filter*'

Specifies the significant portion of the issuer's distinguished name used as a filter. The value specified for IDNFILTR must begin with a prefix found in the following list, followed by an equal sign (X'7E'). Each component should be separated by a period (X'4B'). The case, blanks, and punctuation displayed when the digital certificate information is listed must be maintained in the IDNFILTR. Since digital certificates only contain characters available in the ASCII character set, the same characters should be used for the IDNFILTR value. Valid prefixes are:

- COUNTRY (specified as C=)
- STATE/PROVINCE (specified as ST=)
- LOCALITY (specified as L=)
- ORGANIZATION (specified as O=)
- ORGANIZATIONAL UNIT (specified as OU=)
- TITLE (specified as T=)
- COMMON NAME (specified as CN=)
- DOMAIN COMPONENT (specified as DC=)
- POSTAL CODE (specified as PC=)
- EMAIL (specified as E=)
- STREET NAME (specified as STREET=)
- USERID (specified as UID=)

This keyword is used with:

- The commands ADDTO, LIST, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Example: IDNFILTR keyword

In this example, users who enter the system with a certificate subject's distinguished name that starts with 'OU=NJ.OU=Sales.O=ABC Co' are assigned acid NJDEPT1 if the certificate was issued by the VeriSign certificate authority.

```
TSS ADDTO(NJDEPT1) CERTMAP(NJMAP1)
                   IDNFILTR('OU=VeriSIgn Class 1 Individual subscriber.
                            O=Verisign,Inc.
                            L=Internet')
                   SDNFILTR('OU=NJ.OU=Sales.O=ABC Co')
```

IESFL1 Keyword—Assign Attributes Interactive User Interface

Valid on z/VSE.

Use the IESFL1 keyword to assign attributes to an interactive user interface. These attributes include:

- Submit to batch authority
- PRIMARY sub-library assignment
- POWER and ICCF delete confirmation
- Alter/allocate VSAM resource authorization.

These attributes correspond to user profile data stored in the z/VSE Control File.

This keyword has the following format:

```
TSS ADDTO(acid) IESFL1(BAT,PSL,COD,VSAM)
```

BAT

Authorizes the IUI to submit jobs to the z/VSE batch environment.

PSL

Assigns the IUI a primary sublibrary named PRIMARY acid.

COD

Requires the IUI to give confirmation when deleting POWER jobs or ICCF members.

VSAM

Authorizes the IUI to:

- Define files, libraries, and alternate indexes
- Process catalogs
- Define and delete VSAM space

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, and REPLACE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: IESFL1 keyword

This example gives the IUI authority to submit jobs to z/VSE batch, and to require delete confirmation for POWER jobs and ICCF members:

```
TSS ADDTO(SYSA) IESFL1(BAT,COD)
```

This example removes the above attributes:

```
TSS REMOVE(SYSA) IESFL1(BAT,COD)
```

IESFL2 Keyword—Assign Interactive Interface User Attributes

Valid on z/VSE.

Use the IESFL2 keyword to assign attributes to an interactive interface user.

These attributes correspond to user profile data stored in the z/VSE Control File.

This keyword has the following format:

```
TSS ADDTO(acid) IESFL2(BQA,ESC,COU,CMD,OLPD,XRM)
```

BQA

Authorizes the IUI to manage all z/VSE POWER jobs.

ESC

Permits the IUI to escape to a native CICS session.

COU

Displays all messages on the console for the IUI. Without this attribute, shows only messages related to this user.

CMD

Authorizes the IUI to get a master console and enter all commands.

OLPD

Authorizes the IUI to delete OLPD incident records.

XRM

Authorizes the IUI to maintain resource profiles within the Interactive Interface. These resource profiles include the user, application, and selection profiles.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, and REPLACE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: IESFL2 keyword

This example gives the IUI authority to maintain Interactive Interface profiles and to permit escape to a native CICS session:

```
TSS ADDTO(SYSA) IESFL2(ESC,XRM)
```

This example removes the above attributes:

```
TSS REMOVE(SYSA) IESFL2(ESC,XRM)
```

IESINIT Keyword—Assign Interactive Interface User Name

Valid on z/VSE.

Use the IESINIT keyword to assign an application profile name initiated when the interactive interface user signs on. This attribute corresponds to user profile data stored in the z/VSE Control File.

This keyword has the following format:

```
TSS ADDTO(acid) IESINIT(profname)
```

The following operands can be used with the IESINIT keyword:

profname

A profile name. It should match an existing profile name defined to the interactive interface.

Range: Up to 8 characters

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, and REPLACE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: IESINIT keyword

This example assigns a profile name of IESEADM initiated when the IUI signs on:

```
TSS ADDTO(SYSA) IESINIT(IESEADM)
```

This example removes the above attribute:

```
TSS REMOVE(SYSA) IESINIT
```

IESSYNM Keyword—Interactive User Interface Synonyms Model

Valid on z/VSE.

Use the IESSYNM keyword to assign an interactive user interface ID used as a model for synonyms. This attribute corresponds to user profile data stored in the z/VSE Control File.

This keyword has the following format:

```
TSS ADDTO(acid) IESSYNM(userid)
```

userid

A user ID. It should match an existing user ID defined to the Interactive Interface.

Range: Up to 8 characters

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, and REPLACE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: IESSYNM keyword

This example assigns a user ID of OPER as a model for synonyms:

```
TSS ADDTO(SYSA) IESSYNM(OPER)
```

This example removes the above attribute:

```
TSS REMOVE(SYSA) IESSYNM
```

IESTYPE Keyword—Assign User Type to IUI

Valid on z/VSE.

Use the IESTYPE keyword to assign a user type to an Interactive user Interface. Also used to assign attributes for new item display and the initial action profile type. These attributes correspond to the user profile data stored in the z/VSE Control File.

This keyword has the following format:

```
TSS ADDTO(acid) IESTYPE(USERTYPEx,NEW,SELECT)
```

USERTYPEx

Defines the overall user characteristics. Replace x with one of the following values:

- 1—z/VSE system administrator (includes access to ICCF)
- 2—Programmer/Operator (includes access to ICCF)
- 3—General application user

NEW

System displays news items to the IUI.

SELECT

Interprets initial action name as a selection profile. If not specified, interprets the action as an application profile.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, and REPLACE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: IESTYPE keyword

This example makes the ACID SYSA a system administrator, have news items displayed, and the initial action a selection profile:

```
TSS ADDTO(SYSA) IESTYPE(USERTYPE1,NEW,SELECT)
```

This example removes the above attributes:

```
TSS REMOVE(SYSA) IESTYPE(USERTYPE1,NEW,SELECT)
```

IESVCAT Keyword—Assign Default VSAM User Catalog

Valid on z/VSE.

Use IESVCAT keyword to assign a default VSAM user catalog for an Interactive User Interface user. This attribute corresponds to user profile data stored in the z/VSE Control File.

This keyword has the following format:

```
TSS ADDTO(acid) IESVCAT(usercat)
```

usercat

A VSAM user catalog name. It should match an existing VSAM user catalog definition.

Range: 1 to 7 characters

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, and REPLACE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: IESVCAT keyword

This example assigns VSESPUC as the default VSAM catalog for the current IUI dialogs:

```
TSS ADDTO(SYSA) IESVCAT(VSESPUC)
```

This example removes the above attribute:

```
TSS REMOVE(SYSA) IESVCAT
```

IMPORT Keyword—Import Certificate from PKCS#11 Token

Valid on z/OS.

Use the IMPORT keyword with the P11TOKEN function to import a certificate and its private key from a PKCS#11 token and add it to CA Top Secret.

This keyword has the following format:

```
TSS P11TOKEN IMPORT LABLCTKN(token name)
             TOKNDATA(userid,digicert)
             SEQNUM(nnnnnnnn)
             [LABLPKDS(PKDS—label—name/*)]
             [WITHLABL(certificate label)]
             [PCICC]
             [ICSF]
```

LABLCTKN(*token name*)

Specifies the name of the token. The token must already exist.

TOKNDATA(*userid,digicert*)

Userid specifies the ACID for the digital certificate. Digicert identifies the digital certificate.

SEQNUM(*nnnnnnnn*)

Specifies the sequence number of the certificate being imported.

LABLPKDS(*PKDS—label—name/)**

(Optional) Specifies the PKDS label of the record created in the ICSF Public Key Data Set (PKDS). The field can be used with the ICSF, PCICC, NISTECC, or BPECC, but many of these keywords cannot be used together (see individual keyword descriptions for details). If neither ICSF or PCICC is specified, a PCICC key is generated by the hardware and saved in CRT format in the ICSF PKDS. If NISTECC or BPECC is specified, an ECC key is generated, otherwise an RSA key is generated.

Specify (*) to take the value from the LABLCERT keyword. In that case, LABLCERT is specified along side LABLPKDS(*). If LABLPKDS(*) is specified without the LABLCERT keyword, an error message is displayed.

In either case, the PKDS label must conform to ICSF label syntax rules. The first character must be alphabetic or national. The field is folded to uppercase.

Valid characters: Alphanumeric, national (@,#,$) or period(.).

Limits: Up to 64 characters

WITHLABL(*certificate label*)

(Optional) Specifies the label to be associated with the imported certificate. If not specified CA Top Secret generates a label name.

Range: Up to 32 Characters

PCICC

(Optional) Specifies that the key pair is generated using the PCI Cryptographic Coprocessor and that the private key is stored in ICSF PKDS.

ICSF

(Optional) Specifies that the generated private key is stored in ICSF.

The administrator must have the following authority:

- MISC3(PTOK).
- MISC4(CERTSITE) when the certificate is from the CERTSITE ACID
- MISC4(CERTAUTH) when the certificate is from the CERTAUTH ACID
- Controlled by ICSF using resources in the CRYPTOZ and IBMFAC resource classes.

To add an imported certificate to CA Top Secret, the authority required for:

- One's own certificate is sufficient authority to CRYPTOZ resources and READ authority to IRR.DIGTCERT.ADD
- Someone else's certificate is sufficient authority to CRYPTOZ resources and UPDATE authority to IRR.DIGTCERT.ADD
- CERTSITE or CERTAUTH certificate is sufficient authority to CRYPTOZ resources and CONTROL authority to IRR.DIGTCERT.ADD

IMSMSC Keyword—Determine MSC Security Level

Valid on z/OS.

Use the IMSMSC keyword to determine the level of security in effect for inbound transactions in an IMS Multiple Systems Coupling (MSC) environment. IMSMSC determines if transaction validation is performed and against what ACID.

Up to 20 MSC linknames can specified per TSS command.

This keyword has the following format:

```
TSS ADDTO(region acid) IMSMSC('MSC linkname(acid|+USER|+DEFAULT),...')
```

region acid

ACID associated with the IMS control region started task.

MSC linkname

The label on the MSNAME macro of the IMS Stage 1 Generation, associated with the incoming MSC transaction. Specify a linkname of ALL for all the previously unspecified linknames in the Stage 1 gen for this region to receive the associated link option.

null-string

Indicates that no validation is performed

acid

Specifies the ACID used for validation. Ensure that the acid specified is not used for session signon (for example, through SOURCE restriction). An ACID used for an IMSMSC link and online session signon can have unpredictable effects.

+USER

Specifies that the userid signed on in the originating IMS region is propagated to the MSC receiving region. Ensure that the session user is allowed to sign on in both the originating and receiving regions, and that the transaction is allowed in both regions.

+DEFAULT

Specifies that the DEFACID assigned by the receiving region FACILITY control option is used for MSC transaction validation for the MSC linkname.

This keyword can be used with:

- The commands ADDTO, REMOVE, and REPLACE
- The ACID type User
- ACID(MAINTAIN) authority

Examples: IMSMSC keyword

This example associates region ACID IMS81 with:

- Explicit link name MSC1$2 with propagated user ID's from the originating region's terminal session
- Explicit link name MSC4$3 with fixed ACID LINK4$3
- Explicit link name MSC9$8 with the receiving region's facility DEFACID
- Implicit assignment of BADACID unable to sign on in the receiving region to any other link not explicitly defined in previously encountered links

```
TSS ADD(IMS81) IMSMSC('MSC1$2(+USER), MSC4$3(LINK4$3), MSC9$8(+DEFAULT),
                      ALL(BADACID)')
```

INCLUDE Keyword—Include Dates

Valid on z/OS, z/VM, and z/VSE.

Use the INCLUDE keyword to add, remove, or replace the list of dates that are specifically included in the calendar record in the SDT Record.

Note: The format of the date in INCLUDE is not affected by the DATE control option.

This keyword has the following format:

```
TSS ADD(SDT) CALENDAR(cal-name)
             [YEAR(yyyy)]
             [DAYS(days)]
             [INCLUDE(mm/dd,...]
             [EXCLUDE(mm/dd,...)]
```

mm/dd

Lists specific dates that are included in addition to the ones specified with the DAYS keyword, or excluded with EXCLUDE.

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The SDT Record exclusively
- MISC3(SDT) authority

Example: INCLUDE keyword

This example creates a calendar for the present year with April 22 and April 29 enabled:

```
TSS ADDTO(SDT) CALENDAR(CAL1)
               INCLUDE(04/22,04/29)
```

INSTDATA Keyword—Record ACID Information

Valid on z/OS, z/VSE, and z/VM.

Use the INSTDATA keyword to record or remove information about an ACID. Up to 255 characters of information about an associated ACID may be used for convenient record keeping, or for interrogation by a user-written Installation Exit.

This keyword has the following format:

```
TSS ADDTO(acid) INSTDATA(['])installation data['])
```

```
TSS REPLACE(acid) INSTDATA(['])installation data['])
```

When used with TSS ADD, CA Top Secret adds installation data to the end of any existing data. The new data is concatenated, but cannot extend beyond 255 characters.

When used with TSS REPLACE, CA Top Secret replaces any installation data. The new data is limited to 255 characters.

Installation data:

- Must be delimited by single quotes if it contains blanks, lower case or special characters
- Is automatically uppercased unless OPTIONS(73 or 16) were set during TSS startup

This keyword is used with:

- The commands CREATE, REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, MSCA, DEPARTMENT, DIVISION, and ZONE
- MISC1(INSTDATA) authority

Examples: INSTDATA keyword

This example adds INSTDATA to ACID QO66:

```
TSS ADDTO(QO66) INSTDATA('JUNE 90')
```

'JUNE 90', is added to the end of the INSTDATA which exists for QO66.

Administrators may remove the entire INSTDATA field for an ACID, but cannot remove specific portions.

This example removes ALL INSTDATA for QO66.

```
TSS REMOVE(QO66) INSTDATA
```

Note: Use the TSS REPLACE (INSTDATA) to totally replace old data with new data.

INTO Keyword—Store TSS Commands to a PDS Data Set

Use the INTO keyword on a LIST with ARCHIVE keyword or a MODEL command to write the specified user's security record into a PDS data set in the form of TSS commands. The produced TSS commands can be used to recreate the user in the future.

Use the ARCHIVE keyword for the DELETE command to archive the user prior to deletion. If ARCHIVE is specified on the DELETE command the INTO keyword is mandatory.

For LIST and DELETE commands, if the INTO keyword is specified the ARCHIVE keyword is also required.

For the INTO keyword:

- The dataset must be a PDS dataset with LRECL=80.
- If the PDS data set specified on the INTO keyword is under TSS protection, the user executing the command must have permission to the data set. If user not authorized:
  - When running in FAIL mode, an error message is displayed and the command is failed
  - When running WARN mode, an error message is issued and the command runs to completion.
- If the PDS dataset specified on the INTO keyword exists, that dataset will be overwritten.
- If the member specified on the INTO keyword already exists, that member will be overwritten.
- If the PDS data set specified on the INTO keyword does not exist, a PDS data set will be created with DCB info (FORMAT=FB, LRECL=80, BLKSIZE= 27920, DSORG=PO, TRKS=14, DIR=5). This new new PDS data set is created on a common storage volume that is defined in the VATLSTnn member of SYS1.PARMLIB.

The first record in theTSS command output is a comment record that identifies the command, the user(s) involved, the date and time of command execution, which TSS administrator issued the command, and what system the command was executed on:

```
/*ARCHIVE  CASSIE   STORED 02/17/10-09.54.00 BY MASTER1  ON XE15
/*MODEL  NEWCAS   STORED 02/18/10-14.41.53 BY MASTER1  ON XE15 USING CASSIE
```

This keyword has the following format:

```
TSS LIST(acid) ARCHIVE INTO(datasetname(membername))
TSS DELETE(acid) ARCHIVE INTO(datasetname(membername))
TSS MODEL USING(acid) ACID(newacid) INTO(datasetname(membername))
```

INTO(*datasetname*(*membername*))

(Optional) Identifies the name and optionally a member name of a PDS data set where commands will be stored. If this keyword is not used for MODEL and LIST commands, TSS commands will be displayed.

datasetname

Specifies the data set where commands will be stored.

Limits: PDS dataset with LRECL=80.

membername

(Optional) Specifies the member name of the data set where commands will be stored.

Default: Value specified in the ACID keyword.

This keyword is used with:

- The commands LIST, DELETE, and MODEL
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority DATA(ALL)

Examples: INTO keyword

Examples using ARCHIVE:

- This example generates the archive command output in a user-specified member in a PDS dataset called KOTPA01.RACHAEL.ARCHIVE with member name Rachaelk:

```
TSS LIST(Rachael) ARCHIVE INTO(kotpa01.rachael.archive(Rachaelk))
```

- This example generates the archive command output in a user specified member in a PDS dataset called KOTPA01.RACHAEL.ARCHIVE without member name. The member will be set to the acid name specified by the LIST keyword:

```
TSS LIST(Rachael) ARCHIVE INTO(kotpa01.rachael.archive)
```

- This example deletes the user Rachael, and generates the archive command output in a user specified member in PDS dataset called KOTPA01.RACHAEL.ARCHIVE without member name:

```
TSS DELETE(Rachael) ARCHIVE INTO(kotpa01.rachael.archive)
```

- This example deletes the user Rachael, and generates the archive command output in a user specified member in PDS dataset called KOTPA01.RACHAEL.ARCHIVE with member name Rachaelk:

```
TSS DELETE(Rachael) ARCHIVE INTO(kotpa01.rachael.archive(Rachaelk))
```

Examples using MODEL:

- This example generates the model command output for new acid NewRac using the existing acid Rachael in a user specified member in a PDS data set called KOTPA01.RACHAEL.MODEL with the member name Rachaelk:

  ```
  TSS MODEL USING(Rachael) ACID(NewRac) INTO(kotpa01.rachael.model(Rachaelk))
  ```

- This example generates the model command output for new acid NewRac using the existing acid Rachel in PDS data set called KOTPA01.RACHAEL.MODEL without a member name. The member name will be set to the acid name specified in ACID keyword:

  ```
  TSS MODEL USING(Rachael) ACID(NewRac) INTO(kotpa01.rachael.model)
  ```

ISSUERDN and SERIALNUM Keywords—Identify Certificate

Valid on z/OS.

Use the ISSUERDN and SERIALNUM keywords to identify the digital certificate for the following transactions:

- Displaying digital certificate information
- Changing TRUST/NOTRUST status
- Removing a user's digital certificate

To identify the certificate you want to update, specify its:

- DIGICERT name
- LABLCERT name
- ISSUERDN and SERIALNUM. The ISSUERDN keyword is always used with the SERIALNUM keyword. ISSUERDN must be in single quotes.

This keyword has the following format:

```
TSS LIST(ACID) SERIALNUM(serial-number)
               ISSUERDN('issuer's-dist-name')
```

This keyword is used with:

- The commands LIST, REPLACE, and REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority for users
- MISC4(CERTSITE) authority for CERTSITE ACIDs
- MISC4(CERTAUTH) authority for CERTAUTH ACIDs

Examples: ISSUERDN and SERIALNUM keyword

This example lists the contents of a digital certificate using the SERIALNUM/ ISSUERDN:

```
TSS LIST(user1) SERIALNUM(5C072BEA12340U)
                ISSUERDN('OU=Class 1 Certificate.
                          O=BobsCertAuth, Inc.
                          L=internet.
                          C=US')
```

This example updates the TRUST status to trusted:

```
TSS REPLACE(acid1) SERIALNUM(5C072BEA12340U)
                   ISSUERDN('Certificate.O=BobsCertAuth, Inc.L=internet.C=US')
                   TRUST
```

This example removes a user's digital certificate:

```
TSS REMOVE(user1) SERIALNUM(5C072BEA12340U)
                  ISSUERDN('Certificate.O=BobsCertAuth, Inc.L=internet.C=US')
```

If the certificate being removed has a connection to a key ring, the certificate is removed along with any key ring connections it may have.

JOBNAME Keyword—Bring Data Set into Hyperspace

Valid on z/OS.

Use the JOBNAME keyword with the DLF ACID to allow a specific data set brought into hyperspace if accessed by one of the jobs in the JOBNAME list. The JOBNAME keyword must be used with the DSNAME resource. Up to five JOBNAMEs can be specified per TSS command.

This keyword has the following format:

```
TSS ADDTO(DLF) DSNAME(SYS1.)
               JOBNAME(job1,job2,...)
```

This keyword is used with:

- The commands ADDTO, CREATE, PERMIT, REFRESH, REMOVE, and REVOKE
- The DLF ACID only
- MISC2(DLF) authority

Examples: JOBNAME keyword

This example, if DLF is authorized, adds JOB1 and JOB2 to an existing DLF authorization. If DLF is not authorized, this example adds the DSNAME to the DLF Record with JOB1 and JOB2 as authorized JOBNAME:

```
TSS ADDTO(DLF) DSNAME(CICS.MASTER.FILE)
               JOBNAME(JOB1,JOB2)
```

This example leaves the data set authorized for DLF, but removes JOB2 from the list:

```
TSS REMOVE(DLF) DSNAME(CICS.MASTER.FILE)
                JOBNAME(JOB2)
```

KERBLINK Keyword—Define Foreign Kerberos Users

Valid on z/OS.

Use the KERBLINK keyword to:

- Define Kerberos foreign principal users to the Secureway Server Network Authentication Service
- Map foreign principal names to CA Top Secret user IDs on your local z/OS system by defining an entry in the KERBLINK SDT record

You can map each principal in a foreign realm to its own user ID on your local z/OS system, or you can map all principals in a foreign realm to the same user ID on your system.

User IDs that map to foreign principals do not require KERB segments.

To define *foreign-principal-name* create an SDT KERBLINK record**.** The KERBNAME contains the principal name, fully qualified with the name of the foreign realm.

```
/…/foreign_realm/[foreign-principal-name]
```

To map a unique CA Top Secret User ID to each foreign principal, specify the foreign realm name and the foreign principal name. To map the same CA Top Secret User ID to every foreign principal in the foreign realm, specify the foreign realm name. In each case, you specify the local User ID using the KERBUSER option.

This keyword has the following format:

```
TSS ADDTO(SDT) KERBLINK(link_name)
               LINKNAME(fully–qualified–name)
               KERBUSER(local_acid)
```

link_name

Identifies the record. Must be a unique name within the KERBLINK SDT record type.

Range: 1 to 8 alphanumeric characters

fully-qualified-name

A string which specifies the URL of the foreign realm in which the foreign principal user is defined. The KERBNAME under which the associated (foreign) ACID is defined. The format of this string is:

```
/ /foreign_realm_URL/Ýforeign_principal_KERBNAME
```

Note: If the foreign_principal_KERBNAME is not supplied, the definition refers to all Kerberos principal users defined in that specific foreign realm.

local_acid

The ACID in the local system to which requested activities is assigned in the local system.

This keyword is used with:

- The commands ADDTO, DELETE, REMOVE, REPLACE, and LIST
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- The SDT Record and on the definition of Kerberos Foreign Principal Users
- MISC3(SDT) authority to ADDTO or REMOVE records or REPLACE fields in the SDT
- MISC8(LISTSDT) authority to list realms in the SDT
- ACID(MAINTAIN) and MISC4(KERBUSER) authority

Examples: KERBLINK keyword

This example lists all KERBLINK records in the SDT:

```
TSS LIST(SDT) KERBLINK(ALL)
```

This example authenticates the KERBNAME('kal-el') defined on foreign URL www.krypton.org and assigns that KERBNAME to local acid CKENT01:

```
TSS ADD(SDT) KERBLINK(SUPERMAN)
             LINKNAME('/…/www.krypton.org/kal–el')
             KERBUSER(CKENT01)
```

This example assigns all Kerberos principal users from the same URL to CKENT01:

```
TSS ADD(SDT) KERBLINK(SUPERMAN)
             LINKNAME('/…/www.krypton.org/')
             KERBUSER(CKENT01)
```

This example maps USER01 and USER02 foreign principal names to their individual user IDs on the local z/OS system:

```
TSS ADDTO(SDT) KERBLINK(KERBLK1)
               LINKNAME('/…/KERB.CA.COM/USER01')
               KERBUSER(PAUL01)
TSS ADDTO(SDT) KERBLINK(KERBLK2)
               LINKNAME('/…/KERB.CA.COM/USER02')
               KERBUSER(NADIA01)
```

This example maps other foreign principals presenting tickets from the KERB.CA.COM server to the PUBLIC01 user ID on the local z/OS system:

```
TSS ADDTO(SDT) KERBLINK(KERBLK3)
               LINKNAME('/…/KERB.CA.COM/')
               KERBUSER(PUBLIC01)
```

KERBNAME Keyword—Specify Local Principals as Users

Valid on z/OS.

Use the KERBNAME keyword to specify local principals as CA Top Secret users. The KERBNAME principal name, which must be added to an existing user, creates a KERB segment in the user record in the Security File. Each local principal must have a password (do not use NOPW option).

For each local principal you define (KERB segment in user record), CA Top Secret automatically creates a mapping entry in the KERBLINK SDT record. If you remove the KERB segment, or delete the user, the KERBLINK record is deleted automatically.

Each local principal must have a key registered with the local Kerberos Server to be recognized as a local principal. The key is generated from the principal's CA Top Secret user password at the time of the user's password change. The user's definition is not complete until the key is generated.

```
TSS ADDTO(acid) KERBNAME(' kerberos–principal–name')
                [ENCRYPT('[DES|NODES]
                         [DES3|NODES3]
                         [DESD|NODESD],
                         [AES128|NOAES128]
                         [AES256|NOAES256]')]
                [MAXTKTLF( max–ticket–life)]
```

kerberos-principal-name

Specifies the z/OS user ID's local *kerberos-principal-name.* The value specified must be unique, therefore, a list of users cannot be specified on an ADDUSER command with the KERBNAME keyword.

The *kerberos—principal—name* defined to CA Top Secret can consist of any character except @ (X'7C'). To avoid problems with different code pages do not use EBCDIC variant characters. Lower case characters are honored.

Note: To use the USS kinit -s command the KERBNAME and the ACID must be identical.

Use the single quotes, or not, depending on the following:

- If parentheses, commas, blanks, or semicolons are entered as part of the name, the name must be enclosed in single quotes.
- If a single quote is intended to be part of the name and the entire character string is enclosed in single quotes, use two single quotes together to represent each single quote with the string.
- If the first character of the name is a single quote, enter the string within single quotes, with two single quotes entered for the single quote.

Examples: KERBNAME keyword

This example indicates that a KERB segment was added to user Cassie, with a Kerberos principal name of Cassie1Principal and a maximum ticket life of 24 hours:

```
TSS ADDTO(CASSIE) KERBNAME('CASSIE1PRINCIPAL')
                  MAXTKTLF(86400)
```

This example lists the KERB segment of ACID CASSIE:

```
TSS LIST(KRBPEON) SEG(KERB)
```

KERBPASS Keyword—Foreign Password

Valid on z/OS.

Use the KERBPASS keyword as a password which must be supplied by a foreign system when the network authentication service connection is initiated.

Each system (A and B) must define itself as a local realm with a KERBPASS (X and Y):

LOCAL REALM A KERBPASS: X	LOCAL REALM B KERBPASS: Y
FOREIGN REALM B KERBPASS: Y	FOREIGN REALM A KERBPASS: X

In addition, to establish a connection, each system must define the corresponding foreign realm with passwords which match their local definition. Defining matched passwords helps ensure authentication at the REALM level.

This keyword has the following format:

```
TSS ADDTO(SDT) REALM(KERBDFLT|foreign_realm)
               REALMNAME(realmname)
               ENCRYPT('[DES|NODES]
                        [DES3|NODES3]
                        [DESD|NODESD]
                        [AES128|NOAES128]
                        [AES256|NOAES256]')
               KERBPASS(password)
               CHKADDRS
```

Note: The CHKADDRS keyword is used only with KERBDFLT realm record, not foreign realms.

This keyword is used with:

- The commands ADDTO and REPLACE
- The SDT Record exclusively
- MISC3(SDT) authority to ADDTO or REMOVE records or REPLACE fields in the SDT
- MISC8(LISTSDT) authority to list realms in the SDT

Example: KERBPASS keyword

This example creates the local realm associated with the URL HYPOTHETICAL.CA.COM enabling all encryption types and a password of "THET1CL".

```
TSS ADD(SDT) REALM(KERBDFLT)
             REALMNAME(HYPOTHETICAL.CA.COM)
             ENCRYPT('DES DESD') (THET1CAL)
             CHKADDRS
```

At a communicating node with URL HONEYPOT.CLIENT1.COM, this same node could be described as a foreign REALM at HYPOTHETICAL.CA.COM with the identical KERBPASS:

```
TSS ADD(SDT) REALM(HYPOTHET)
             REALMNAME('/.../HONEYPOT.CLIENT1.COM/krbtgt/HYPOTHETICAL.CA.COM')
             ENCRYPT('DES DESD')
             KERBPASS(THET1CAL)
```

KERBPASS Keyword—REALM Record Password

Valid on z/OS.

Use the KERBPASS keyword to specify the value of the REALM record password and is applicable to all REALM (local and foreign) SDT definitions.

A KERBPASS password is required for the realm to grant ticket-granting-tickets. This password must be associated with the definition of an inter-realm trust relationship, or the definition is incomplete.

This keyword has the following format: - Local Realm

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME('kerberos-realm-name')
               MINTKTLF(min-ticket-life)
               MAXTKTLF(max-ticket-life)
               DEFTKTLF(default-ticket-life)
               KERBPASS(kerberos-password)
               CHKADDRS
```

This keyword has the following format: - Foreign Realm

```
TSS ADDTO(SDT) REALM(realm-label)
               REALMNAME('fully-qualified-name')
               KERBPASS(password)
```

kerberos-password

Specifies the value of the Kerberos password in the local realm.

Maximum length: 8 characters

password

Specifies the value of the Kerberos password in the foreign realm.

Maximum length: 8 characters

You can use any character, but do not use any of the variant characters to avoid problems with different code pages.

Use the single quotes in the following circumstances:

- If parentheses, commas, blanks, or semicolons are entered as part of the name, the character string must be enclosed in single quotes.
- If a single quote is part of the name and the entire character string is enclosed in single quotes, use two single quotes together to represent each single quote in the string.
- If the first character of the name is a single quote, enter the string within single quotes with two single quotes entered for the single quote.

Both uppercase and lowercase characters are accepted and maintained in the case entered.

This keyword is used with:

- The commands ADDTO, DELETE, REMOVE, REPLACE, and LIST
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- MISC3(SDT) authority to ADDTO or REMOVE records or fields in the SDT
- MISC8(LISTSDT) authority to list KERBNAME in the SDT

Examples: KERBPASS keyword

This example indicates that a KERBPASS of children was used in the default REALM SDT record:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME(LOCAL.CA.COM)
               MINTKTLF(30)
               MAXTKTLF(86400)
               DEFTKTLF(36000)
               KERBPASS(CHILDREN)
               CHKADDRS
```

This example indicates that a KERBPASS XXXXXXXX was used for a foreign realm SDT record: (must be the same as local KERBPASS)

```
TSS ADDTO(SDT) REALM(KERBFOR1)
               REALMNAME('LOCAL.CA.COM/KRBTGT/FOREIGN.SERVER_1')
               KERBPASS(XXXXXXXX)
```

KERBSEGM Keyword—Kerberos Principal User Record

Valid on z/OS.

KERBSEGM is an SDT record automatically created when a Kerberos principal user is defined. It may not be manipulated by commands to the SDT, except under the supervision of CA Top Secret Support. However, the values may be listed for documentation of problems with the Secureway Server Network Authorization Service.

This keyword has the following format:

```
TSS LIST(SDT) KERBSEGM(acid|ALL)
```

acid

Represents an ACID defined to CA Top Secret to which a KERBNAME has been added.

ALL

All records of type KERBSEGM are listed.

This keyword used with:

- This LIST command
- The SDT Record exclusively
- MISC8(LISTSDT) authority

Example: KERBSEGM keyword

This example lists all Kerberos principal acids in the SDT:

```
TSS LIST(SDT) KERBSEGM(ALL)
```

KERBUSER Keyword—Map Foreign Principal Names

Valid on z/OS.

Use the KERBUSER keyword to map foreign principal names to individual user IDs.

To map a unique CA Top Secret User ID to each foreign principal, specify the foreign realm name and the foreign principal name. To map the same CA Top Secret User ID to every foreign principal in the foreign realm, you need only specify the foreign realm name. In each case, you specify the local User ID using the KERBUSER option.

This keyword has the following format:

```
TSS ADDTO(SDT) KERBLINK(label-name)
               LINKNAME('fully-qualified-name')
               KERBUSER(userid)
```

userid

Specifies the User ID on the local z/OS system.

This keyword is used with:

- The commands ADDTO, DELETE, REMOVE, REPLACE, and LIST
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(KERBUSER) authority

Example: KERBUSER keyword

This example maps USER's foreign principal name to the individual user ID on the local z/OS system:

```
TSS ADDTO(SDT) KERBLINK(KERBLK1)
               LINKNAME('KERB.CA.COM/USER01')
               KERBUSER(PAUL01)
```

KERBVIO Keyword—Count Attempts to Use a Key

Valid on z/OS.

Use the KERBVIO keyword to count unsuccessful attempts to use a Network Authentication Service key. This count is incremented when an incorrect password is entered in response to *a kinit* function. When KERBVIO contains a value, which exceeds the CA Top Secret PTHRESH, the associated acid is suspended, and the value of KERBVIO is reset to zero. Manipulation of this field should be handled entirely by requests to the Kerberos server. In cases where corruption has made internal processing impossible to reset, the field can be removed from the associated acid and the system will automatically re-initialize it to zero on the next *kinit* function.

Important! Do not ADD or REPLACE the value of KERBVIO. This has unpredictable effects.

This keyword has the following format:

```
TSS REMOVE(acid) KERBVIO
```

acid

Specifies the accessor ID of the user whose KERBVIO value is being reset.

This keyword is used with:

- The REMOVE command
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority
- Administrators who do not have ACID(MAINTAIN) authority can remove the KERBVIO field for users within their scope if given UPDATE access to entity TSSCMD.USER.REMOVE.KERBVIO in the CASECAUT resource class

Example: KERBVIO keyword

This example resets the value of KERBVIO to zero on the acid KRBPEON:

```
TSS REMOVE(KRBPEON) KERBVIO
```

KEYRING Keyword—Add a Key Ring to a User

Valid on z/OS.

Use the KEYRING keyword to:

- Add a key ring to an individual user. A user can be a member of more than one key ring. The ring name is unique within the user, and the name you specify in KEYRING identifies the key ring for a user.
- List information about a keyring.

When used with ADDTO, this keyword has the following format:

```
TSS ADDTO(acid) KEYRING(8-byte-ring-name)
```

When used with LIST, this keyword has the following format:

```
TSS LIST(ACIDS) KEYRING(ALL)
```

For each certificate connected to the key ring, the following information is displayed:

- Name of the certificate
- Owner of the certificate
- Label assigned to the certificate
- DEFAULT status of the certificate within the ring
- Usage within the ring

Key rings can be listed for ACID types user, CERTAUTH, and CERTSITE.

This keyword is used with:

- The commands ADDTO, REMOVE, REPLACE, and LIST
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- MISC8 ACID(MAINTAIN), MISC4(CERTUSER), and MISC4(CERTLIST) authority
- Administrators who do not have the above authority need:
  - UPDATE access to TSSCMD.CERTUSER.ADDTO in the CASECAUT resource class to ADD a KEYRING keyword
  - UPDATE access to TSSCMD.CERTUSER.REMOVE in the CASECAUT resource class to REMOVE a KEYRING keyword
  - UPDATE access to TSSCMD.CERTUSER.REPLACE in the CASECAUT resource class to REPLACE a KEYRING keyword

Examples: KEYRING keyword

This example adds a key ring to a user:

```
TSS ADDTO(USER01) KEYRING(CASRING)
```

This example identifies the key ring by its ring name or by its label:

```
TSS LIST(acid) {KEYRING(8-byte ring name)}
               {LABLRING(237-byte ring label)}
```

This example lists all the ACIDs and the key rings associated with them:

```
TSS LIST(ACIDS) KEYRING(ALL)
```

This example lists information on a specific key ring:

```
TSS LIST(acid) LABLRING(name)
```

This example displays all key rings associated with a particular user:

```
TSS LIST(user01) KEYRING(ALL)
```

KEYSIZE Keyword—Specify Key Size

Valid on z/OS.

Use the KEYSIZE keyword to specify the size of the private encryption key in decimal bits.

This keyword has the following format:

`TSS REKEY(acid) KEYSIZE(`*nnn*`n)`

KEYSIZE

- Specifies the size of the private encryption key in decimal bits.
- **Limits:**
  - Non-NISTECC and non-BPECC: any numeric value from 512 to 4096.
  - For NISTECC, one of the following: 192, 224, 256, 384, 521.
  - For BPECC, one of the following: 160, 192, 224, 256, 320, 384, 512.
- **Default:**
  - Non-NISTECC and non-BPECC:1024.
  - NISTECC and BPECC: 192.

The maximum key size is dependent on the private key type.

Private key type maximum key sizes are:

- BPECC key—512 bits
- NISTECC key—521 bits
- ICSF RSA key—1024 bits
- DSA key —2048 bits
- Non-ICSF RSA key—4096 bits
- PCI-class cryptographic coprocessor RSA key—4096 bits

Shorter ECC keys have key strengths comparable to longer RSA keys. The following table displays the comparable strength of each key type:

RSA Key Size (in bits)	NISTECC Key Size (in bits)	BPECC Key Size (in bits)
1024	192	160 or 192
2048	224	224
3072	256	256 or 320

RSA Key Size (in bits)	NISTECC Key Size (in bits)	BPECC Key Size (in bits)
7680	384	384
15360	521	512

Currently, the standard key sizes for RSA keys are as follows:

- 512—Specifies a low-strength key
- 1024—Specifies a medium-strength key
- 2048—Specifies a high-strength key
- 4096—Specifies a very high-strength key

This keyword is used with:

- The commands GENCERT and REKEY
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority for users
- MISC4(CERTSITE) authority for CERTSITE ACIDs
- MISC4(CERTAUTH) authority for CERTAUTH ACIDs

Example: KEYSIZE keyword

This example creates low strength key:

```
TSS REKEY(user1) DIGICERT(cert0001)
                 NEWDIGIC(cert0002)
                 KEYSIZE(512)
```

KEYSMSTR Keyword—Define Password Key Name

Valid on z/OS and z/VM.

Use the KEYSMSTR keyword:

- In conjunction with DCENCRY to provide a key name to encrypt and decrypt passwords.
- (On z/OS) To have the SMB server use encrypted password processing, add the entry DCE.PASSWORD.KEY to the SDT KEYSMSTR record.

To use the EIM/PROXY feature, add the KEYSMSTR entry LDAP.BINDPW.KEY before entering a PRXBINDPW.

This keyword has the following format:

```
TSS ADD(SDT) KEYSMSTR(LDAP.BINDPW.KEY
             DCENCRY(CCCCCCCCCCCCCCCC)
             [KEYMASK|KEYENCRY]
```

DCENCRY

A hexadecimal encryption key value.

Size: 16 characters

KEYMASK

(Default) Indicates that the DCENCRY key is used to mask the user's DCE password when it is stored in the DCEKEY field of the user's acid record.

KEYENCRY

Indicates that the DCENCRY key is used to encrypt the user's DCE password when it is stored in the user's acid record.

This keyword is used with:

- The commands ADDTO, DELETE, REMOVE, REPLACE, and LIST
- The SDT records exclusively
- MSCA authority

Examples: KEYSMSTR keyword

This example defines the string C1C2C3C4C5C6C7C8 as the encryption key value for the LDAP PROXY BINDPW key:

```
TSS ADD(SDT) KEYSMSTR(LDAP.BINDPW.KEY)
             DCENCRY(C1C2C3C4C5C6C7C8)
```

This example lists the KEYSMSTR record:

```
TSS LIST(SDT) KEYSMSTR(LDAP.BINDPW.KEY)
```

This example deletes the KEYSMSTR from the SDT:

```
TSS DELETE(SDT) KEYSMSTR(LDAP.BINDPW.KEY)
```

KEYUSAGE—Specify Key Usage Extension

Valid on z/OS

Use the KEYUSAGE keyword to specify the appropriate values for the KeyUsage certificate extension, of which one or more of the values might be coded. For certificate authority certificates (CERTAUTH), the default is CERTSIGN and is always set. There is no default for certificates that are not certificate-authority certificates. When the key pair is generated using the DSA algorithm, only the digital signature bit is set since the keys cannot be used for encryption.

This keyword has the following format:

```
TSS GENCERT(acid) KEYUSAGE(HANDSHAKE|DATAENCRYPT|DOCSIGN|CERTSIGN|KEYAGREE)
```

HANDSHAKE

Facilitates identification and key exchange during security handshakes, such as SSL, which set the digitalSignature and keyEncipherment indicators. When the key pair is generated using the DSA algorithm, only the digitalSignature bit is set because the keys cannot be used for encryption.

DATAENCRYPT

Encrypts data, which sets the dataEncipherment indicator. When the key pair is generated using the DSA algorithm, you cannot use the DATAENCRYPT keyword in the Keyusage parameter.

DOCSIGN

Specifies a legally binding signature, which sets the nonRepudiation indicator.

CERTSIGN

Specifies a signature for other digital certificates and CRLs, which sets the keyCertSign and cRLSign indicators.

KEYAGREE

Facilitates key exchange, which sets the keyAgreement indicator. This usage is valid only for NISTECC and BPECC keys. A certificate with no keyUsage value other than keyAgreement cannot be used for signing.

Note: Include single quotes if specifying more than one value with KEYUSAGE. For example:

```
KEYUSAGE('HANDSHAKE DATAENCRYPT')
```

This keyword is used with:

- The GENCERT and RENEW commands
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority for users

- MISC4(CERTSITE) authority for CERTSITE ACIDs
- MISC4(CERTAUTH) authority for CERTAUTH ACIDs

LABLCERT Keyword—Specify Certificate Label

Valid on z/OS.

Use the LABLCERT keyword to specify a label associated with a certificate.

Up to 32 case-sensitive-characters can be specified for the label name. Spaces are allowed if you use single quotes.

This label is a descriptive identifier for a certificate and must be unique for the individual user. If a label is not specified, the label field defaults to the value specified in the DIGICERT keyword.

The keywords DIGICERT and DSDCN are mandatory. If the value specified in DIGICERT or LABLCERT already exists for this user, a message indicates this and the certificate is not added.

If the user did not specify LABLCERT, and the data set being processed is PKCS#12, the label is extracted from the PKCS#12 package and truncated to 32 characters, if required. If LABLCERT is not specified, or extracted from a PKCS#12 package, CA Top Secret uses the certificate name specified by keyword DIGICERT as the label for the certificate.

If DIGICERT and LABLCERT are specified, DIGICERT takes precedence over LABLCERT, except in ADDTO transactions.

This keyword has the following format:

```
TSS ADDTO(acid) DIGICERT(name)
                DCDSN(dsname)
                LABLCERT(label name)
```

This keyword is used with:

- The commands ADDTO, EXPORT, GENCERT, GENREQ, LIST, P11TOKEN, REMOVE, RENEW and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority for users
- MISC4(CERTSITE) authority for CERTSITE ACIDs
- ACID(MAINTAIN) and MISC4(CERTAUTH) authority

Example: LABLCERT keyword

This example specifies LABLCERT label name when adding a user:

```
TSS ADDTO(USER01) DIGICERT(CERT0001)
                  DCDSN(USER1.CERT01)
                  LABLCERT(CASLABL000123)
```

This example uses a label as a handle instead of the serial number and issuer's distinguished name, and must be unique for the individual user:

```
TSS ADDTO(acid) DIGICERT(name) DCDSN(dsname)
                [START(sdate)]
                [FOR(date)|UNTIL(date)]
                [LABLCERT('label name')]
                [TRUST|NOTRUST|HIGHTRUST]
```

LABLCMAP Keyword—Specify Certificate Filter Label

Valid on z/OS.

Use the LABLCMAP keyword to specify the label associated with the digital certificate filter.

This keyword has the following format:

```
TSS ADDTO(acid) CERTMAP(recid)
                SDNFILTR('subject-dist-name-filter')
                IDNFILTR('issuer-dist-name-filter')
                LABLCMAP('32-byte label')
```

The label name must be enclosed in single quotation marks.

Up to 32 characters can be specified for the label name. It can contain embedded blanks and mixed-case characters, and it is stripped of leading and trailing blanks. If a single quotation mark is intended to be part of the label name, use two single quotation marks together for each single quotation mark within the string, and the entire string must be enclosed within single quotation marks.

LABLCMAP is optional on the TSS ADDTO CERTMAP command. If it not included on the command, the record identifier name entered for CERTMAP is automatically used for LABLCMAP.

This keyword is used with:

- The commands ADDTO, LIST, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Example: LABLCMAP keyword

This example adds the certificate filter NJMAP1, with a subject's distinguished name filter of OU=NJ.OU=Sales.O=ABC Co., to user NJDEPT1, and gives it the label name of NJ DEPT 1 Map:

```
TSS ADDTO(NJDEPT1) CERTMAP(NJMAP1)
                   SDNFILTR('OU=NJ.OU=Sales.O=ABC Co')
                   LABLCMAP('NJ DEPT 1 Map')
```

LABLPKDS Keyword—Specify Certificate Label

Valid on z/OS.

Use the LABLPKDS keyword to specify an optional label associated with the certificate private or public key being stored into the ICSF storage facility. Up to 64 characters can be specified for the label name. The field can be used with the ICSF, PCICC, NISTECC, or BPECC keywords.

This label is used as a descriptive identifier for a certificate ICSF key, and must be unique across the MVS complex.

This keyword has the following format:

```
TSS ADDTO(acid) LABLPKDS(PKDS–label–name/*)

LABLPKDS(PKDS–label–name/*)
```

LABLPKDS(*PKDS—label—name/)**

(Optional) Specifies the PKDS label of the record created in the ICSF Public Key Data Set (PKDS). The field can be used with the ICSF, PCICC, NISTECC, or BPECC, but many of these keywords cannot be used together (see individual keyword descriptions for details). If neither ICSF or PCICC is specified, a PCICC key is generated by the hardware and saved in CRT format in the ICSF PKDS. If NISTECC or BPECC is specified, an ECC key is generated, otherwise an RSA key is generated.

Specify (*) to take the value from the LABLCERT keyword. In that case, LABLCERT is specified along side LABLPKDS(*). If LABLPKDS(*) is specified without the LABLCERT keyword, an error message is displayed.

In either case, the PKDS label must conform to ICSF label syntax rules. The first character must be alphabetic or national. The field is folded to uppercase.

Valid characters: Alphanumeric, national (@,#,$) or period(.).

Limits: Up to 64 characters

This keyword is used with:

- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- The commands GENCERT, ADDTO, P11TOKEN, and REKEY
- ACID(MAINTAIN) for users within their scope, and MISC4(CERTGEN)
- MISC4(CERTSITE) for SITE
- ACID and MISC4(CERTAUTH) for CERTAUTH ACID.

Examples: LABLPKDS keyword

This example adds a digital certificate to the security file and the private key is stored in the ICSF PKDS data set. The value for the PKDS label is user1.cert0002.

```
TSS ADD(user1) DIGICERT(cert0001)
               DCDSN(user.certificate.pfx12) pkcspass(password)
               LABLPKDS(user1.cert0001)
               ICSF
```

This example adds a digital certificate with the private key stored in ICSF. The value for the PKDS label is taken from the LABLCERT keyword. The certificate label will be user.certificate.personel. The PKDS label will be USER.CERTIFICATE.PERSONEL.

```
TSS ADD(user1) DIGICERT(cert0002)
               DCDSN(user.certificate.pfx12) pkcspass(password)
               LABLPKDS(*)
               ICSF
               LABLCERT(user1.cert0002.personel)
```

This example generates an ECC certificate with the private key stored in the ICSF PKDS. The value for the PKDS label is USER1.CERT0003.BPECC.

```
TSS GENCERT(user1) DIGICERT(cert0003)
               LABLPKDS(user1.cert0003.bpecc)
               PCICC
               BPECC
```

This example generates an ECC certificate with the private key stored in the ICSF PKDS. The value for the PKDS label is USER1.CERT0004.NISTECC.

```
TSS GENCERT(user1) DIGICERT(cert0004)
               LABLPKDS(user1.cert0004.nistecc)
               NISTECC
```

LABLRING Keyword—Key Ring Label

Valid on z/OS.

Use the LABLRING keyword to specify a 237 byte label to associate with the key ring. This label must be unique to the key ring, because it is used to identify the key ring. This field is optional.

If not specified, the 8-byte KEYRING name is automatically added to the LABLRING.

If KEYRING is specified with LABLRING, KEYRING takes precedence, except in ADDTO transactions.

This keyword has the following format:

```
TSS LIST(acid) [KEYRING(8-byte ring name)]
               [LABLRING(237-byte ring label)]
```

This keyword is used with:

- The commands ADDTO, REMOVE, LIST, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTUSER) authority

Examples: LABLRING keyword

This example adds a key ring to a user's ACID record:

```
TSS ADDTO(acid) KEYRING(8-byte key ring name)
               [LABLRING(237-byte ring name)]
```

This example adds a key ring with a label name:

```
TSS ADDTO(USERA) KEYRING(USERARNG)
                 LABLRING('USERA KEYRING FOR z/OS')
```

This example replaces a label name:

```
TSS REPLACE(USERA) KEYRING(USERARNG)
                   LABLRING('USERA KEYRING FOR z/OS')
```

LANGUAGE Keyword—Specify Language Preference

Valid on z/OS, z/VSE, and z/VM.

Use the LANGUAGE keyword to assign or remove a language preference code, which is passed to the message processing Installation Exit.

This keyword has the following format:

```
TSS ADDTO(acid) LANGUAGE(c)
```

c

Must equal a user-defined, one-character language preference code.

Capacity of list

One LANGUAGE code per CA Top Secret command.

The site must write an installation exit for message translation to allow association of the language preference code with the language of preference. CA Top Secret passes this code to the installation exit for message translation.

This keyword is used with:

- The commands CREATE, REPLACE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Examples: LANGUAGE keyword

This example indicate that error messages are in French (and the administrator has defined F as the code for French):

```
TSS ADDTO(userque) LANGUAGE(F)
```

This example removes the language preference:

```
TSS REMOVE(userque) LANGUAGE(F)
```

LDAPDEST Keyword—Define Node to LDAP Node List

Valid on z/OS.

Use the LDAPDEST keyword to add, remove, or replace nodes to the LDAP node list of an ACID record. The LDAPDEST node list determines which LDAP nodes will receive updates performed on valid ACIDs. Up to 5 node names can be added by a single ADD command. Specifying a null LDAPDEST list on a REMOVE, removes all currently defined LDAP nodes for the ACID.

To assign default remote node IDs for an ACID. An ACID's DEFNODES list will only be searched if the CPFTARGET control option is set to AUTO and no TARGET has been specified on the command.

When used with ADDTO, this keyword has the following format:

```
TSS ADDTO(acid_name) LDAPDEST(node,node,,)
```

When used with REMOVE, this keyword has the following format:

```
TSS REMOVE(acid_name) LDAPDEST(node,node,,)
```

When used with REPLACE, this keyword has the following format:

```
TSS REPLACE(acid_name) LDAPDEST(node,node,,)
```

This keyword is used with:

- The commands ADD, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(DEFNODES) authority

LDAPNODE Keyword—Define LDAP Nodes

Valid on z/OS.

Use the LDAPNODE keyword to define LDAP nodes to the CA Top Secret database as NDT node elements.

The keyword is used with:

- The commands ADD, LIST, REMOVE, and REPLACE
- The NDT record
- MISC2(NDT) authority

This keyword has the following format:

```
TSS ADDTO(NDT) LDAPNODE(node_name)
               ACTIVE(YES|NO)
               ADMDN(LDAP administrator_distinguished_name)
               ADMPSWD(LDAP_administrator_password)
               APPLNAME(application_name)
               BITDEFLT(bit_field_format)
               BROADCAST(YES|NO)
               CHILDELETE(YES|NO)
               CODEPAGE(encoding_table)
               DATEFMT(date_format)
               DEBUG(YES|NO)
               EXTENDED(YES|NO)
               LABLCERT(label_name)
               PSWDLOWR(YES|NO)
               USERDNS(distinguished_name_suffix)
               JOURNAL(YES|NO)
               RECOVERY(YES|NO)
               SYNCADD(YES|NO)
               SYNCDEL(YES|NO)
               SYNCUPD(YES|NO)
               OBJCLASS(LDAP_object_class)
               PSWDASIS(YES|NO)
               SYSID(sysid1,sysid2,,,)
               URL(Uniform_Resource_Locator)
               XREF(ACIDfield1,LDAPattribute1Name,LDAPattribute1FieldType,
                    LDAPattribute1DataFormat,LDAPattribute1Length,
                    EncloseCharacter)
```

LDAPNODE(*node_name*)

(Required) Specifies the internal NODE name of the LDAP server.

SYSID(*sysid1,,,*)

Specifies up to five SMF IDs of systems where the LDAPNODE definition apply. The SYSID value might contain an asterisk for masking. If you omit SYSID, the LDAPNODE is global for all systems sharing the security file. You can define more than five system IDs by using multiple ADD commands. When specified on an ADD command, the new SYSID replaces a previously existing value.

ACTIVE(YES|NO)

Specifies whether the node is active and attempting to communicate with the specified LDAP server.

Default: NO

ADMDN(*LDAP_administrator_distinguished_name*)

(Required) Specifies the LDAP administrator distinguished name that is used for binding to the LDAP server and administering the LDAP request. The following representations indicate that string substitutions are allowed for this field:

- TSS administrator's name (20 bytes) from the ACID record
- TSS administrator's LID (8 bytes) from the ACID record

If any embedded spaces or commas exist within the ADMNDN specification, enclose the entire string in quotation marks. For example, enter a ADMNDN value like this:

```
ADMDN('cn=%L, o= CAI, c=USA')
```

Note: You must specify the LDAP administrator password or the APPLNAME field. Each LDAP request requires an administrator distinguished name and a password. To provide the password, you can specify the administrator password in the NDT LDAPNODE element. Alternatively, you can identify the TSS administrator and the NDT table data record used for generating PassTickets, by specifying the APPLNAME in the NDT LDAPNODE element. The two methods are mutually exclusive.

When updating an existing LDAPNODE definition, you can modify the password or the application name without specifying the ADMINDN keyword.

ADMPSWD(*LDAP_administrator_password*)

Specifies the LDAP administrator password that is used in conjunction with the LDAP administrator ID for binding to the LDAP server.

APPLNAME(*application_name*)

Specifies the application name of the NDT table data record that contains the TSS administrator's encryption key that is used for generating PassTickets. The TSS administrator's userid is used in conjunction with the PassTicket for binding to the LDAP server and administering the LDAP request.

Note: For information about NDT records, see the *CA Top Secret User Guide*.

BITDEFLT(*field_type*/*format*)

Specifies the default field type and format to use for all bit fields that are sent to the LDAP server. The default for this field is CHAR_YN. The following options are available for this field:

BINARY

Binary 1 or 0 when the bit is ON or OFF, respectively.

CHAR_01

'1' or '0' when the bit is ON or OFF, respectively.

CHAR_YN

'Y' or 'N' when the bit is ON or OFF, respectively.

CHAR_TF

'T' or 'F' when the bit is ON or OFF, respectively.

BINARY_REV

Binary 0 or 1 when the bit is ON or OFF, respectively.

CHAR_REV01

'0' or '1' when the bit is ON or OFF, respectively.

CHAR_REVYN

'N' or 'Y' when the bit is ON or OFF, respectively.

CHAR_REVTF

'F' or 'T' when the bit is ON or OFF, respectively.

Note: This default can be overridden per bit field specified in the XREF parameter.

BROADCAST(YES|NO)

Specifies whether the node is a broadcast node, which means that all commands and password changes are sent to this node regardless of the LDS attribute setting on the ACID record.

Default: NO

SYNCADD(YES|NO)

Specifies whether TSS ACID create processing is propagated to the LDAP server.

Default: NO

CHILDELETE(YES|NO)

Specifies whether to delete children objects before deleting the base object.

CODEPAGE(*character*)

Specifies a 20-byte identifier of the character encoding table to use to translate characters as they pass into the system. If no CODEPAGE is specified, ASCII ISO8859-1 is assumed.

DATEFMT(*date_format*)

Specifies the default format for the date fields that are sent to the LDAP server. Available date format options are MMDDYYYY, DDMMYYYY, YYYYMMDD, MMDDYY1, DDMMYY1, and YYMMDD1. MM represents a two-digit month, DD represents a two-digit day, and YYYY represents a four-digit year. YY represents a two-digit year, and 1 represents a / delimiter in the date field. The default date format is MMDDYYYY. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069).

Note: This default can be overridden per date field specified in the XREF parameter.

DEBUG(YES|NO)

Specifies whether to enable or disable node level tracing.

EXTENDED(YES|NO)

Specifies whether to use extended operations to enable SSL for the connection to the LDAP server.

LABLCERT(*label_name*)

Defines the LABEL of the PERSONAL certificate used, if CLIENT authentication is required for the LDAP server defined by this LDAPNODE record.

PSWDLOWR(YES|NO)

Specifies whether to propagate the case sensitivity format of the user's password.

PSWDLOWR works in conjunction with the PSWDASIS function as follows:

- If PSWDASIS(NO) and PSWDLOWR(YES) exist, the password is sent in lowercase.
- If PSWDASIS(YES) and PSWDLOWR(YES) exist, PSWDASIS overrides PSWDLOWR, which means that passwords are sent as is.
- If PSWDASIS(NO) and PSWDLOWR(NO) exist, the password is sent in uppercase.

When a user changes a password during system entry validation, LDAP Directory Services (LDS) automatically propagates the new password to the LDAP servers that are interested in the password field. The user receives no indication that LDS processing was involved. LDS must be active, and the LDS option must be specified in the ACID.

Note: For more information about LDS, see the *CA Top Secret User Guide*.

Default: NO

USERDNS(*distinguished_name_suffix*)

Indicates the user distinguished name suffix that refers to the entry on the LDAP server where the changes are applied. This field has a maximum length of 255 characters. The following representations indicate that string substitutions are allowed for this field:

- N—User's Name
- L—User's ACID

For example, in USERDNS ('tssacid=N%, host=prod, o=company, c=usa') %N substitutes the TSS user's name (20 bytes) from the user's ACID record.

In USERDNS ('tssacid=USER, host=prod, o=company, c=usa'), USER substitutes the TSS user's ACID (8 bytes) from the user's ACID record.

Note: If embedded spaces or commas exist within the USERDNS specification, enclose the entire string in quotation marks. For example, you might enter a USERDNS value like this:

```
USERDNS('o= CAI, ou=Development Team, c=USA')
```

JOURNAL(YES|NO)

Specifies whether to enable journaling of LDAP outbound traffic.

OBJCLASS(*LDAP_object_class*)

(Required) Specifies the LDAP object class to use when an LDAP entry is created. The object class defines the attributes that the LDAP directory entry might contain. The default object class is TSSUSER.

SYNCDEL(YES|NO)

Specifies whether to propagate TSS ACID remove processing to the LDAP server.

Default: NO

SYNCUPD(YES|NO)

Specifies whether to propagate TSS ACID add/rep processing to the LDAP server.

Default: NO

RECOVERY(YES|NO)

Specifies whether to enable recovery processing for the node.

Default: YES

PSWDASIS(YES|NO)

(If the password field is specified in the XREF field) Specifies whether to propagate the password as it was entered during a signon password change. Any changes made to the password through the TSS command always propagate in uppercase even if this option is YES. If this option is set to NO, signon password changes propagate in uppercase.

Default: YES

URL(*Uniform_Resource_Locator*)

(Required) Specifies the Uniform Resource Locator (URL) that identifies the LDAP server. There is a maximum of three URL entries. The entries specify the primary followed by the backups. The syntax of the LDAP URL is as follows:

```
ldap[s]:// [<host>[:CA Portal]]
```

ldap

Specifies a connection using the LDAP protocol.

ldaps

Specifies an SSL LDAP connection.

host

Specifies the name or IP address of the LDAP server host.

port

Specifies the port number of the LDAP server.

XREF(ACIDfield,LDAPattributeName,LDAPattributeFieldType, LDAPattributeDataFormat,LDAPattributeLength,EncloseCharacter)

(Required) Specifies the names of the TSS ACID fields and the corresponding LDAP directory attribute fields synchronized to the LDAP directory. The following parameters are required:

ACIDfield

Specifies any ACID related keyword that can be specified on a TSS ADD/REP/CRE command for a user type ACID.

LDAPattributeName

Specifies the name of the LDAP directory attribute.

The following optional XREF parameters might be specified to override the default format of DATE and BIT fields:

LDAPattributeFieldType

Specifies the field type of the LDAP attribute. Valid field types are BIT, DATE, and UNICODE. If not specified, the default is the TSS ACID field type from which the LDAP attribute has been mapped. If you specify this parameter, you must also specify LDAPattributeDataFormat.

LDAPattributeDataFormat

Specifies the data format of the LDAP attribute. For DATE type, valid data formats are the same as for the DATEFMT field. For Unicode, valid formats are UTF16LE, UTF16BE, UTF32LE, and UTF32BE. For BIT type, valid data formats are the same as for the BITDEFLT field, as follows:

```
LDAP          BIT  BIT
Attribute     is   is
Data Format   ON   OFF

CHAR_YN      'Y'   'N'
CHAR_REVYN   'N'   'Y'

CHAR_TF      'T'   'F'
CHAR_REVTF   'F'   'T'

CHAR_01      '0'   '1'
CHAR_REV01   '1'   '0'

BINARY        x'0   x'1
BINARY_REV    x'1   x'0
```

LDAPattributeLength

Specifies a number that represents the maximum data length of the LDAP attribute. If you omit this parameter specification, the default length is the TSS ACID field length from which the LDAP attribute has been mapped.

Range: 1 to 1024

EncloseCharacter

(Optional) Specifies to surround the LDAP attribute value with the character represented by the enclose character identifier. This parameter is used with LDAP functions that require particular data values to be surrounded by quotation marks. Valid values are as follows:

SQ

Uses single quotation marks to surround the LDAP attribute value (for example, 'Jay').

DQ

Uses double quotation marks to surround the LDAP attribute value (for example, “Jay”).

Note: By default, the LDAP attribute value is not enclosed by any character.

Notes:

- Optional fields are positional and must be delimited by a comma.
- If the LDAPNODE xref password field has a special LDAPattributetype of UNICODE and LDAPattributedataformat of UTF16LE, UTF16BE, UTF32LE, or UTF32BE, this specification translates the password field into the corresponding UTF selection. UNICODE is used in conjunction with Windows AD only. The XREF fields allow any Unicode specified for all fields; however, this results in an error. If a corresponding LDAPattributedataformat is not specified when UNICODE is entered for the LDAPattributetype, the default is UTF16LE.

Example: Create an LDAP Node with a Single XREF Subfield

This example creates a new LDAP node named “testnode” with a single XREF subfield:

```
TSS ADD(NDT) LDAPNODE(testnode)
             ADMDN('cn=USER1, o=CAI, c=USA')
             ADMPSWD(password)
             USERDNS('o=CAI, ou=TSS Team, c=USA')
             URL(ldap://ca.ldap.server:7000)
             XREF(ACIDNAME,ldap_attr_name1)
```

Example: Manipulate XREF Subfields for an Existing LDAPNODE Entry

This example adds or modifies XREF subfields for an existing LDAPNODE entry:

```
TSS ADD(NDT) LDAPNODE(testnode)
             XREF(DEPT,ldap_attr_name2)
```

Example: Remove XREF Subfields

This example removes XREF sub fields:

```
TSS REM(NDT) LDAPNODE(testnode)
             XREF(DEPT,ldap_attr_name2)
```

Example: Delete a LDAPNODE Definition

This example deletes an entire LDAPNODE definition:

```
TSS REM(NDT) LDAPNODE(testnode)
```

Example: Display an LDAPNODE Definition

This example displays an LDAPNODE definition:

```
TSS LIST(NDT) LDAPNODE(ALL|testnode)
```

Example: Replace an LDAPNODE Definition

This example replaces an LDAPNODE definition:

```
TSS REP(NDT) LDAPNODE(testnode)
             ADMDN('cn=USER1, o=CAI, c=USA')
             ADMPSWD(password)
             USERDNS('o=CAI, ou=TSS Team, c=USA')
             URL(ldap://ca.ldap.server:7000)
             XREF(DEPT,ldap_attr_name2)
```

The REP command removes all XREF entries

LDS Keyword—Add or Remove LDS Attribute

Valid on z/OS.

Use the LDS keyword to add or remove the LDS attribute to or from an ACID record. All entry methods are accepted for the LDS command.

Enabling the LDS option starts up the TSSLDS server address space. Disabling the LDS option terminates the TSSLDS server address space.

This keyword has the following format:

```
TSS ADDTO|REMOVE(acid) LDS
```

This keyword is used with:

- The commands ADD, CREATE, and REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Examples: LDS keyword

This example makes user USER001 valid for LDAP synchronization:

```
TSS ADD(user001) LDS
```

This example removes the LDS attribute:

```
TSS REM(user001) LDS
```

LDSYSID Keyword—Define LDS Global Options

Valid on z/OS.

Use the LDSYSID keyword to define LDS global options to the TSS database as NDT LDSYSID elements.

This keyword has the following format:

```
TSS ADDTO(NDT) LDSYSID(system smfid)
               DEBUG(YES/NO)
               RETRY(nnn)
               TIMEOUT(nnn)
               JOURNAL(YES/NO)
               JOURNALDSN(dsname)
               KEYRING(ring_name)
```

LDSYSID(smfid)

The SMFID of the system where the LDS global options apply.

DEBUG(YES/NO)

Indicates that global tracing of all LDAP nodes is enabled or disabled.

JOURNAL(YES|NO)

Specifies whether journaling of all LDAP outbound traffic is enabled

JOURNALDSN(dsname)

Specifies the Dataset name of the file used for writing journal records when LDS journaling is enabled.

RETRY(nnn)

Specifies the number of times the LDS server task will retry a failed send operation to the remote LDAP directory, before deactivating the LDAP node.

Default: 3

TIMEOUT(nnn)

Specifies time interval in seconds at which the LDS server task will stop waiting for a response from the remote LDAP directory and schedule a retry attempt for the stalled send operation.

Default: 5 seconds

KEYRING(ringname)

Specifies the name of the SSL keyring holding the digital certificates which are used by LDS for SSL authentication.

The keyword is used with the commands ADD, LIST, REMOVE, and REPLACE.

LIBRARY Keyword—Specify Privileged Program Library

Valid on z/OS.

Use the LIBRARY keyword to specify libraries or library prefixes in which a privileged program must reside. LIBRARY is only valid when specified with the permission of DSNAMEs.

If PRIVPGM is not specified with LIBRARY, then the executing program must come from a link listed library or the link pack area (LPA). The program cannot be loaded from a JOBLIB, STEPLIB, or TASKLIB.

If PRIVPGM is specified with LIBRARY, then the executing program must come from the library specified in the PERMIT command function.

Although no specific authority is required, DSNAME(XAUTH) authority to specify LIBRARY for data sets that are owned.

This keyword is used with the PERMIT command.

This keyword has the following format:

```
TSS PERMIT(acid) DSNAME(p-fix)
                 PRIVPGM(p-fix)
                 LIBRARY(p-fix)
```

Prefix length

Two to 44 characters

Capacity of list

One to three prefixes or names per CA Top Secret command

The library specified in the command must be one of the following types:

- Job library as referenced via the //JOBLIB DD statement
- Step library as referenced via the //STEPLIB DD statement
- Task library such as that used with TSO CALL

Example: LIBRARY keyword

This example permits use of production data sets, but only if program PN077 is used for access and the program comes from a specific library:

```
TSS PERMIT(HTEE4) DSNAME(PROD)
                  PRIVPGM(PN077)
                  LIBRARY('PROD.DLIB')
```

When a program resides in both the LINKLIST and the TASKLIB and both are referenced by a job, two PERMITs are necessary.

This example PERMITs the use of data sets when program PN077 is in the LINKLIST and in a TASK LIBRARY named PROD.DLIB:

```
TSS PERMIT(user) DSNAME('PROD')
                 ACCESS(READ)
                 PRIVPGM(PN077)

TSS PERMIT(user) DSNAME('PROD')
                 ACCESS(READ)
                 PRIVPGM(PN077)
                 LIBRARY('PROD.DLIB')
```

LINKID Keyword—Identify LUs for APPC Conversation

Valid on z/OS, ESA 4.2.0 and above only.

Use the LINKID keyword to identify which LUs can be used for APPC conversation processing.

This keyword has the following format:

```
TSS ADDTO(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

The name of the local LU.

remotelu

The partner LU.

Prefixing is supported. Masking is not.

Note: If you have the network qualified names feature active in VTAM, use the four-level name or a prefix. For example:

```
local–netid.luid1.remote–netid.luid2
```

If you do not have the network qualified names feature active in VTAM, use the three-level name. For example:

```
netid.localLU.remoteLU
```

The following keywords are used with LINKID:

SESSKEY

A 16-byte hexadecimal “password” used to verify the link when security is in effect.

INTERVAL

Indicates the number of days for which the SESSKEY is valid. When a value for INTERVAL is not supplied and SESSKEY is supplied, the default for INTERVAL is INTERVAL(0) and the SESSKEY does not expire.

Range: 0 to 32767

CONVSEC

Determines what security information needs to be validated when a conversation request is received. The following operands are used with the CONVSEC keyword:

- NONE—Indicates that no security validation is performed.
- ALREADYV—Indicates that security validation has already occurred. A valid userid and password, however, are still required.
- CONV—Security information is required. A userid and password must be verified.
- PERSISTV—Indicates that a userid and password must be verified on the first request, on subsequent requests only the userid is verified.
- AVPV—Supports both ALREADYV and PERSISTV values. The security type used depends on the incoming request.

SESSLOCK

Indicates that these particular LUs are not authorized to be used for APPC conversations.

The SESSKEY, INTERVAL, and CONVSEC keywords cannot be used with the REMOVE command function. Use the the REPLACE function and specify keyword() to remove the field from the APPCLU Record.

This keyword is used with:

- The commands ADDTO, REMOVE, LIST, and REPLACE
- The APPCLU ACID only
- MISC2(APPCLU) authority

Examples: LINKID keyword

This example establishes 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:

```
TSS ADDTO(APPCLU) LINKID(SYS1.LU01.LU02)
                  CONVSEC(CONV)
                  SESSKEY(1234)
                  INTERVAL(30)
```

This example removes ownership:

```
TSS REMOVE(APPCLU) LINKID(SYS1.LU01.LU02)
```

This example indicate that the SESSKEY provided for the LU01-LU02 link must be changes every 14 days:

```
TSS ADDTO(APPCLU) LINKID(SYS1.LU01.LU02)
                  SESSKEY(1234)
                  INTERVAL(14)
```

LINKNAME Keyword—Specify Foreign Realm

Valid on z/OS.

Use the LINKNAME keyword to specify the fully qualified names of both the foreign realm name and the foreign principal name in a trust relationship.

This keyword has the following format:

```
TSS ADDTO(SDT) KERBLINK(label-name)
               LINKNAME('fully-qualified-name')
               KERBUSER(USERID)
```

fully-qualified-name

Contains the foreign realm name followed the foreign principal name.

This keyword is used with:

- The commands ADDTO, REMOVE, REPLACE, and LIST
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- MISC3(SDT) authority to ADDTO or REMOVE records or fields in the SDT
- MISC8(LISTSDT) authority to list LINKNAME in the SDT

Example: LINKNAME keyword

This example specifies the win2000.ca.server with Jonathan:

```
TSS ADDTO(SDT) KERBLINK(KERBLK1)
               LINKNAME('WIN2000.CA.SERVER/JONATHAN')
               KERBUSER(KERB001)
```

LINUXNAM Keyword—Define Linux User Name

Valid on z/OS.

Use the LINUXNAM keyword to add, remove, or replace the Linux user name to an acid.

This keyword has the following format:

```
TSS ADDTO|REMOVE|REPLACE(acid_name) LINUXNAM(linux_username)
```

LINUXNAM(linux_username)

The Linux user name.

Maximum: 1024 characters.

This keyword is used with:

- The commands ADD, REMOVE, REPLACE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Example: LINUXNAM keyword

This example associates Linux userid LongLINUXuserid1 with the acid LUSER1:

```
TSS ADDTO(LUSER1) LINUXNAM(LongLINUXuserid1)
```

LINUXNODE Keyword—Define LINUX Nodes

Valid on z/OS.

Use the LINUXNODE keyword to add, remove, or replace Linux nodes to the NDT node list of an ACID record.

This keyword has the following format:

```
TSS ADD|REMOVE|REPLACE(NDT) LINUXNODE(node_name)
                            [IPADDR(ip_address)
                            FACILITY(facility_name)
                            ACTIVE(YES|NO)]
```

LINUXNODE(*node_name*)

Linux system name.

Maximum length: 246 characters

IPADDR(ip_address)

IP address for the Linux system.

FACILITY(facility_name)

Facility name to use of the system entry validation

ACTIVE(YES|NO)

Indicates whether the node is active and whether system validation is performed. Valid values are:

- YES—The node is active and system validation is performed.
- NO—(Default) The node is inactive and system validation is not performed.

This keyword is used with:

- The commands ADD, LIST, REMOVE, and REPLACE
- The NDT record only
- MISC2(NDT) authority

Example: LINUXNODE

This example adds the new Linux system name LSERVER with IP address 201.212.0.8, using facility LNXPROD for validations:

```
TSS ADD(NDT) LINUXNODE(LSERVER)
             IPADDR(201.212.0.8)
             FACILITY(LNXPROD)
             ACTIVE(YES)
```

LNXENTS Keyword—Provide Sign on Information

Valid on z/OS and z/VM.

Use the LNXENTS keyword to provide signon information onto an acid for LINUX in addition to the LINUXNAM.

LNXENTS should be added to the acids specified in the OMVSUSR and OMVSGRP control options using the ALL facility. These LNXENTS entries act as a default for any users or groups who do not have LNXENTS specified on their acid. For information, see the OMVSUSR or OMVSGRP control options.

When used with a user, this keyword has the format:

```
TSS ADDTO(acid) LNXENTS(facility,UID|?,home,shell,group acid)
                RANGE(xxx,xxx)
```

When used with a group, this keyword has the format:

```
TSS ADDTO(acid) LNXENTS(facility,GID|?)
                RANGE(xxx,xxx)
```

xxx,xxx

Specifies the range of numbers the UID/GID is selected from. The first number available is selected. If RANGE keyword is not specified, then the first available number starting at 500 is used.

When used with REMOVE, this keyword has the format:

```
TSS REMOVE(acid) LNXENTS(facility)
```

Facility

Multiple LNXENTS records are allowed on an acid, but must have only one defined per facility.

UID

Has the same functionality as the OMVS UID (see UID). This is a positive numeric value. If a user is assigned UID(0), then superuser privileges are assigned. Auto assignment is also specified by a "?" character and will find a number in a given range. The range of the auto assignment can be specified by using the RANGE keyword, or will simply find the first available number starting at 500 if no range is specified (some LINUX systems have the first 500 UIDs pre-assigned).

Range: 0 to 2,147,483,647

Home

A home subdirectory for LINUX. See the HOME keyword for OMVS for more information).

Range: Up to 1024 characters.

Shell

The LINUX equivalent to OMVSPGM for OMVS. See the OMVSPGM keyword for OMVS for more information.

Range: Up to 1024 characters.

Group Acid

The default group acid from which the LINUX GID is used for the user at signon. This group acid must be one of the group acids defined on the user's acid.

When adding/replacing a LNXENTS onto a user acid, all of the fields are required. When adding/replacing onto a group acid, only the facility and the GID are allowed. When performing a remove, only the facility is allowed.

This keyword is used with:

- The commands ADD, REMOVE, REPLACE and WHOHAS
- The ACID types SCA, LSCA, ZCA, VCA, DCA, and User
- ACID(MAINTAIN) authority

Examples: LNXENTS keyword

This example removes the connection between a Linux UID and an ACID on TSO:

```
TSS REMOVE(USER02) LNXENTS(TSO)
```

This example automatically assigns a number to an ACID in any facility using the default range and use john's default group acid:

```
TSS ADDTO(johndoe) LNXENTS(ALL,?,/home/doe,/u,DFLTGRP)
```

This example automatically assign a number to an ACID within a specific range for any facility:

```
TSS ADDTO(johndoe) LNXENTS(ALL,?,/home/doe,/u,DFLTGRP)
                   RANGE(10000,200000)
```

This example displays what LNXENTS numbers are used for a given facility:

```
TSS WHOHAS LNXENTS(TSO,*)
```

LOADLIB Keyword—Specify SIGVER record program library definition

Valid on z/OS.

Use the LOADLIB keyword to define a SIGVER record program entry qualified by a library and volume. When specifying a LOADLIB, you can specify the data set name and volume name (optional).

The LOADLIB keyword is optional when adding a new program to the SIGVER record, but must be specified when administering or listing an existing program which was defined using a LOADLIB qualification.

The name can contain letters, numbers, or special characters.

Keyword length

1-44 characters for the dataset name subfield

6 characters for the volume subfield

Capacity of list

One LOADLIB specification per command.

This keyword is used with:

- The commands LIST, ADDTO, REPLACE and REMOVE
- The SIGVER record only
- MISC5(SGVADMIN) authority to maintain the SIGVER record.
- MISC5(SGVLIST) authority to LIST the SIGVER record.

Examples: LOADLIB keyword

This example adds program PROG01 to the SIGVER Record qualified by data set HLQ.LOADLIB1 on volume VOL001:

```
TSS ADDTO(SIGVER) PROGRAM(prog01) LOADLIB(hlq.loadlib1,vol001)
```

This example removes program PROG01 qualified by data set HLQ.LOADLIB1 on volume VOL001 from the SIGVER Record:

```
TSS REM(SIGVER) PROGRAM(prog01) LOADLIB(hlq.loadlib1,vol001)
```

This example lists the SIGVER entry for program PROG01 qualified by data set HLQ.LOADLIB1 on volume VOL001:

```
TSS LIST(SIGVER) PROGRAM(prog01) LOADLIB(hlq.loadlib1,vol001)
```

LTIME Keyword—Minutes Until Terminal Locks

Valid on z/OS, z/VSE, and z/VM.

Use the LTIME keyword to specify how long (in minutes) until a user's terminal locks if CA Top Secret does not detect activity at that user's terminal.

A user's LTIME overrides the facility LOCKTIME that may be set as a sub-option of the FACILITY control option.

This keyword has the following format:

```
TSS ADDTO(acid) LTIME(nnn)
                [,facility])
```

nnn

Minutes until the terminal locks. (0 = do not lock.)

Range: 0 to 120

This keyword is used with:

- The commands ADDTO, CREATE, REPLACE, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC1(LTIME) authority

Examples: LTIME keyword

This example locks a signed-on user's terminal if it is unused for 15 minutes:

```
TSS ADDTO(QQ35R10) LTIME(15)
```

This example locks the signed-on user's terminal if CICS is unused for 15 minute:

```
TSS ADDTO(QQ35R10) LTIME(15,CICS)
```

This example removes the LTIME value from an ACID:

```
TSS REMOVE(QQ35R10) LTIME(15)
```

This example deactivates automatic terminal locking for an ACID:

```
TSS REMOVE(QQ35R10) LTIME(0)
```

MAPDATA Keyword—Define MAPDATA Field

Valid on z/OS, z/VSE, and z/VM.

Use the MAPDATA keyword to add, remove, or replace a MAPDATA field in the MAP record of the SDT Record. MAPDATA allows the administrator to specify field characteristics associated with MAPREC(map-name).

MAP records are used to support Screen Level Protection (SLP) records.

This keyword has the following format:

```
TSS ADDTO(SDT) MAPREC(map-name)
               MAPDATA(field-definition)
```

field-definition

Specifies the layout of this screen field. You can specify up to five fields on each MAPDATA keyword. Each field-definition operand consists of five sub-fields:

fld-name

Specifies an up to 24-character field name unique to this MAP record. The name can consist of letters, digits, or national characters and can be qualified with periods (.), underscores (_), or hyphens (-).

type

Indicates the field type:

- CHAR—Specifies any of the EBCDIC characters, left-justified and padded with blanks.
- BIN—Specifies up to 16 decimal digits with optional sign. If no sign is specified, a positive value is assumed.
- PACKED—Specifies up to 16 decimal digits with optional sign. If no sign is specified, a positive value is assumed.
- ZONED—Specifies up to 16 decimal digits with optional sign. If no sign is specified, a positive value is assumed.
- HEX—Specifies up to 256 hexadecimal digits (0-F) left-justified and padded with nulls (X"00").

row

Specifies a numeric row for this data field.

column

Specifies a numeric column for this data field.

length

Specifies the length for this data field (optional).

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The SDT Record exclusively
- MISC3(SDT) authority

Examples: MAPDATA keyword

This example adds a MAPDATA field to the MAP record ENG1 that contains a salary field in character format called PAY which is in the fifth row and sixth column, and has a length of eight characters:

```
TSS ADDTO(SDT) MAPREC(ENG1)
               (PAY,CHAR,5,6,8)
```

This example calculates the value for column if the screen image was built by CICS BMS (DFHBMS), by using the value specified in the DFHMDF POS field and adding 1 to the column value. For example if PANA DFHMDF POS=(3,25), length=5:

```
TSS ADDTO(SDT) MAPREC(PANA)
               MAPDATA(CODE,CHAR,3,25,5)
```

If the screen image was not built by CICS BMS (DFHBMS), or you do not have access to the CICS BMS map source, calculate the column by physically counting (from 1) the offset to the field column on the screen.

This example replaces an existing MAP record with one new field:

```
TSS REPLACE(SDT) MAPREC(PANA)
                 MAPDATA(CODE,CHAR,3,15,5).
```

Even though only one field's worth of data is entered with the REPLACE syntax, all previously entered fields are replaced by the single field entered in the REPLACE command. If the MAP record consists of seven fields, all seven original fields are lost and one new field replaces them. There is no command to replace a single field.

MAPREC Keyword—Reference MAP Record

Valid on z/OS, z/VSE, and z/VM.

Use the MAPREC keyword to:

- Provide a name CA Top Secret references a MAP record in the SDT Record by
- To PERMIT or REVOKE a MAP record associated with an OTRAN or PPT resource.

MAP records are used to support SLP records for OTRAN and PPT resources. The screen being protected through SLP is a one-for-one relationship to the transaction (OTRAN) or program (PPT). You can only protect *one* screen per OTRAN or PPT resource.

The administrator can:

- Specify any or all of the following access levels that are associated with OTRANs and PPTs: ALL, INQUIRE, SET, EXECUTE, ALL, NONE, and UPDATE. The only exception for OTRAN access is UPDATE.
- Use any of the following methods to control access to MAP records: Expiration, Facility, Time/Day, and Actions**.**

If ACCESS is not specified, CA Top Secret defaults to EXECUTE access for both OTRAN and PPT.

When used with ADDTO, this keyword has the following format:

```
TSS ADDTO(SDT) MAPREC(map-name)
               DESCRIPT(descript-name)
               MAPDATA(field-definition)
```

When used with PERMIT and REVOKE, this keyword has the following format:

```
TSS PERMIT(acid) [OTRAN(oper)|PPT(oper)]
                  MAPREC(map-name)
                  SELECT(sel-name)
```

Capacity of list

One MAP record per TSS command

map-name

Specifies user-defined record ID that must be unique for each MAP record that can contain letters, numbers, and special characters.

Range: 1 to 8

This keyword is used with:

- This commands ADDTO, DELETE, PERMIT, REMOVE, REPLACE, and LIST
- The SDT Record exclusively
- MISC3(SDT) authority
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA

Examples: MAPREC keyword

This example creates a MAP record called ENG1 in the SDT:

```
TSS ADDTO(SDT) MAPREC(ENG1)
               MAPDATA(PAY,5,6,8)
```

This example grants ALL access to transaction PAYR, provided the selection criteria in SELECT(DEPTREC) yield the value TRUE. The field layout used to interpret this logic is defined in MAPREC(ENG1):

```
TSS PERMIT(USR01) OTRAN(PAYR)
                  ACCESS(ALL)
                  MAPREC(ENG1)
                  SELECT(DEPTREC)
```

This example revokes access:

```
TSS REVOKE(USR01) OTRAN(PAYR)
```

MASKDATA Keyword—SDT Record Mask

Valid on z/OS, VSE, and z/VM.

Use the MASKDATA keyword to add, remove, or replace a MASKDATA field in the MASK record of the SDT Record. MASKDATA allows the administrator to specify field characteristics and a mask-character-string used to overlay the protected field within an associated RECORD.

If using RLP:

- Add the RLP record to the SDT
- Give the RECDATA layout and create the necessary SELECT criteria
- Add the MASK record and MASKDATA to the SDT

This keyword has the following format:

```
TSS ADDTO(SDT) MASKREC(mask-name)
               MASKDATA(field-definition
```

field-definition

Contains the layout of the field to be masked. You can specify up to five fields on each MASKDATA keyword. Each field-definition operand consists of five sub-fields:

fld-name

Specifies a field name unique to this MASK record. The name can consist of letters, digits, or national characters and can be qualified with periods (.), underscores (_), or hyphens (-).

Range: Up to 24 characters

type

Indicates a field type. Replace type with one of these values:

- CHAR—Specifies any of the EBCDIC characters, left-justified and padded with blanks.
- BIN—Specifies up to 16 decimal digits with optional sign. If no sign is specified, a positive value is assumed.
- PACKED—Specifies up to 16 decimal digits with optional sign. If no sign is specified, a positive value is assumed.
- ZONED—Specifies up to 16 decimal digits with optional sign. If no sign is specified, a positive value is assumed.
- HEX—Specifies up to 256 hexadecimal digits (0-F) left-justified and padded with nulls (X"00").

offset

Indicates an up to five-digit initial position (relative to 1) for the field value that must not exceed the record length.

length

Indicates a decimal value for the length of the masked field.

Range: 1 to 44

mask

Indicates a mask value that can contain letters, numbers, and special characters. However, the mask value must match the field's data type. See the table below for specific rules.

Size: Up to 44-character

Note: Do not use quotes to surround the mask unless they are actually included in the mask.

The following table illustrates a mask-field type and its respective mask-field literal type:

Field Type	Literal Character-Set	Left Sign
CHAR	Alphanumeric	No
BIN	{0,1}	Yes
HEX	0-F	No
PACKED	0-9	Yes
ZONED	0-9	Yes

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The SDT Record
- MISC3(SDT) authority

Examples: MASKDATA keyword

This example adds a MASKDATA field to the MASK record CRYPT1 that contains a salary field in character format called ADDR, masked field with an asterisk (*), having an offset value of 50 and a length of 20 characters:

```
TSS ADDTO(SDT) MASKREC(CRYPT1)
               MASKDATA(ADDR,CHAR,50,20,********************)
```

This example replaces an existing MASK record with one new field:

```
TSS REPLACE(SDT) MASKREC(CRYPT1)
                 MASKDATA(ADDR,CHAR,50,10,**********)
```

Even though only one field's worth of data is entered with the REPLACE syntax, all previously entered fields are replaced by the single field entered in the REPLACE command. If the MASK record consists of seven fields, all seven original fields are lost and one new field replaces them. There is no command to replace a single field.

MASTFAC (z/OS and z/VSE) Keyword—Override Default Facility

Valid on z/OS and z/VSE.

Use the MASTFAC keyword to override the default facility, controlled by a batch or started task ACID, associated with a multi-user facility such as IMS or CICS. (By default, CICS uses the CICSPROD facility, and IMS uses the IMSPROD facility. MASTFAC allows use of other defined facilities such as CICSTEST and IMSTEST.)

The following describes when the MASTFAC attribute is valid.

- MASTFAC is only of value to the BATCH or STC ACID which controls the region, not to the ACIDs that sign on.
- MASTFAC is valid only for online multi-user regions such as CICS, IMS, CA-Roscoe, and CA-IDMS.

The facility name used as an operand for MASTFAC must be defined to CA Top Secret in the systems Facilities Matrix.

Additional facilities can be defined via the NAME sub-option of the FACILITY control option. For information, see the *Implementation: CICS, Implementation: IMS, or Implementation: Other Interfaces Guide.*

This keyword has the following format:

```
TSS ADDTO(region acid) MASTFAC(facility)
```

This keyword is used with:

- The commands CREATE, REPLACE, ADDTO, and REMOVE
- The User ACID type
- MISC9(MASTFAC) authority

Examples: MASTFAC keyword

This example assumes that the administrator had already CREATED the batch or started task ACID (CICST1) andindicates that facility CICSTEST is controlled by ACID CICST1:

```
TSS ADDTO(CICST1) MASTFAC(CICSTEST)
```

This example removes the relationship between a region and a facility:

```
TSS REMOVE(CICST1) MASTFAC(CICSTEST)
```

MASTFAC (z/VM) Keyword—Associate Virtual Machine to a Facility

Valid on z/VM.

Use the MASTFAC keyword to associate a virtual machine, which is issuing security validation requests via the Application Interface, to a unique facility.

The facility name used as an operand for MASTFAC must be defined to CA Top Secret in the systems Facilities Matrix. For information on the FACILITY control option, see the *Control Options Guide* .

This keyword has the following format:

```
TSS ADDTO(virtual machine acid) MASTFAC(facility)
```

This keyword is used with:

- The commands CREATE, REPLACE, ADDTO, and REMOVE
- The user ACID type
- MISC9(MASTFAC) authority

Examples: MASTFAC keyword

In this example a disconnected service machine called SERMACH processes requests on behalf of other users. A security interface within SERMACH validates those user requests. This example indicates that security calls originating from the SERMACH machine are processed using a facility called SERFAC:

```
TSS ADDTO(SERMACH) MASTFAC(SERFAC)
```

This example assumes the administrator had created an ACID called SERMACH and had defined a facility called SERFAC within the CA Top Secret Facilities Matrix.

This example removes the MASTFAC keyword from the ACID:

```
TSS REMOVE(SERMACH) MASTFAC(SERFAC)
```

MASKREC Keyword—MASK Record Name

Valid on z/OS, z/VSE, and z/VM.

Use the MASKREC keyword to:

- Provide a name for a MASK record which CA Top Secret uses in the SDT Record
- To PERMIT or REVOKE a MASK record together with a SELECT statement associated with an FCT.

The administrator can:

- Specify the same access levels associated with an FCT resource: SET, INQUIRE, ALL, BROWSE, DELETE, NONE, READ, and UPDATE.
- Use any of the following methods to control access: Expiration, Facility, Time/Day, and Actions.

If ACCESS is not specified, CA Top Secret defaults to READ access.

When used with ADDTO, this keyword has the following format:

```
TSS ADDTO(SDT) MASKREC(mask-name)
               MASKDATA(field-definition)
```

mask-name

Specifies a user-defined record ID that must be unique for each MASK record that can contain letters, numbers, and special characters.

Range: 1 to 8

When used with PERMIT, this keyword has the following format:

```
TSS PERMIT(acid) FCT(oper)
                 ACCESS(access-level)
                 MASKREC(mask-name)
                 SELECT(selin,selout)
```

selin

Specifies the input select record.

selout

Specifies the output select record.

Note: It is not necessary to have both an input and output record for a SELECT statement.

Capacity of list

One MASKREC and SELECT statement per TSS command

This keyword is used with:

- The commands ADDTO, DELETE, LIST, PERMIT, REMOVE, REPLACE, and REVOKE
- The SDT Record exclusively
- MISC3(SDT) authority

Example: MASKREC keyword

This example creates a MASK record called CRYPT1 in the SDT:

```
TSS ADDTO(SDT) MASKREC(CRYPT1)
               MASKDATA(ADD,CHAR,50,20,***************)
```

This example permits a user ALL access to the FCT file called PAY with the MASK record CRYPT1 and selects all departments 100 and above from the input record, and all the employees named Mike from the output record:

```
TSS PERMIT(USR01) FCT(PAY)
                  ACCESS(ALL)
                  MASKREC(CRYPT1)
                  SELECT('IF DEPT GE "100" AND NAME EQ "MIKE" ')
```

This example revoke access:

```
TSS REVOKE(USR01) FCT(PAY)
```

MASKREC Keyword—Overlay FCT Values

Use the MASKREC keyword in conjunction with SELECT and RECORD SDT specifications to overlay the values on file for an FCT and thus provide field-level protection under Record Level Protection (RLP). When MASKREC is supplied for an FCT PERMIT, SELECT is also required. The association with RECORD is implicit by matching the FCT resource-name with the RECORD.

The administrator can:

- Specify the same access levels associated with an FCT resource: SET, INQUIRE, ALL, BROWSE, DELETE, NONE, READ, and UPDATE. If ACCESS is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access: Expiration, Facility, Time/Day, and Actions.

This keyword has the following format for PERMIT/REVOKE:

```
TSS PERMIT(acid) FCT(oper)
                 ACCESS(access-level)
                 MASKREC(mask-name)
                 SELECT(selread)
```

Capacity of list

One MASKREC and SELECT statement per TSS command

mask-name

Specifies the SDT MASKREC name applied to the PERMIT.

selread

Specifies the SDT SELECT name used as the selection process for all file accesses.

This keyword is used with:

- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC3(SDT) authority

Examples: MASKREC keyword

This example gives ALL access on the FCT(PAY) provided that the SELECT logic defined for ISDEPT interpreting field names with RECORD(PAY) is TRUE. The fields defined for MASKREC(CRYPT1) are overlaid with the mask-characters specified to prevent users from seeing these fields.

```
TSS ADDTO(SDT) MASKREC(CRYPT1)
               MASKDATA(PAY,PACKED,110,6,000000)

TSS PERMIT(USR01) FCT(PAY)
                  ACCESS(ALL)
                  MASKREC(CRYPT1)
                  SELECT(ISDEPT)
```

This example revokes access:

```
TSS REVOKE(USR01) FCT(PAY)
```

MATCHLIM Keyword—Limit Audit Activity

Valid on z/OS.

Use the MATCHLIM keyword to limit the number of loggings generated because of resource or user auditing.

When added to an acid, this keyword has the following format:

```
TSS ADDTO(acid) AUDIT MATCHLIM
```

When used with the AUDIT reserved ACID, this keyword has the following format:

```
TSS ADDTO(AUDIT) resclass(resname) MATCHLIM
```

resclass

Specifies the class of the resource that will be added to the AUDIT reserved ACID.

name

Specifies the name of the resource that will be added to the AUDIT reserved ACID.

Notes:

- When adding MATCHLIM to an acid, the AUDIT attribute must be present. AUDIT can already be on the acid or specified on the same command.
- If MATCHLIM is specified for a volume resource in the AUDIT record, all datasets on that volume are subject to match limit processing. Each dataset is tracked separately and logging is discontinued only for datasets that reach the MATCHLIM value.
- Audit Match Limit processing only affects loggings that are generated because of the AUDIT attribute or the AUDIT record. All other loggings, including those that occur because of violations, bypass authorities, and exits, are not affected by Match Limit processing.
- If the match limit is reached for a resource or user, no more loggings are generated because of the AUDIT attribute/record. Loggings will resume if the MATCHLIM(CLEAR) modify command is issued. An IPL will also cause match limit processing to resume for all users and resources if the control option is set by the user or specified in the parm file.
- To permanently disable match limit processing for a user or resource, remove the MATCHLIM keyword from the specific acid or AUDIT record. To disable match limit processing for all auditing, set the MATCHLIM control option to 0.
- If AUDIT is removed from an acid, MATCHLIM is also removed.

For more information about the MATCHLIM control option, see the *Control Options Guide*.

This keyword is used with:

- The commands ADDTO, CREATE, and REMOVE
- The ACID types AUDIT, User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(AUDIT) authority to ADDTO or REMOVE the MATCHLIM attribute for ACIDs within their scope
- RESOURCE(AUDIT) admin authority when maintaining a record on the AUDIT reserved ACID

Examples: MATCHLIM keyword

- This example adds the AUDIT and MATCHLIM attribute to USER01. Any actions performed by USER01 are recorded in the Audit file until the match limit value is reached:

  ```
  TSS ADD(USER01) AUDIT MATCHLIM
  ```

- This example removes the AUDIT and MATCHLIM attribute from USER01. MATCHLIM is automatically removed if AUDIT is removed:

  ```
  TSS REMOVE(USER01) AUDIT
  ```

- This example removes the MATCHLIM attribute from USER01. Auditing continues without match limit processing:

  ```
  TSS REMOVE(USER01) MATCHLIM
  ```

- This example allows activity for the specified resource to be audited until the match limit value is reached:

  ```
  TSS ADD(AUDIT) resclass(resname) MATCHLIM
  ```

MAXLEN Keyword—Define FDT Maximum Length

Valid on z/OS, z/VM, and z/VSE.

Use the MAXLEN keyword to define or change the number of bytes allowed for the user-defined FDT entry.

This keyword has the following format:

```
TSS ADDTO(FDT) MAXLEN(nnnnn)
```

nnnnn

A decimal number,.

Range: 1 and 32767

This keyword is used with:

- The commands ADDTO and REPLACE
- The FDT record
- MISC1(RDT) authority

Example: MAXLEN Keyword

This example changes an FDT field length from 8 to 17:

```
TSS ADDTO(FDT) FDTNAME($PERS)
               MAXLEN(8)
               SEGMENT(PERSDEPT)
               FDTCODE(1E)
               DISPLAY('PERSDEPT')

TSS REPLACE(FDT) FDTNAME($PERS)
                 MAXLEN(17)
```

Note: The total number of bytes available for all user-defined fields is 32,767.

MAXLEN Keyword—Define Maximum Permission Length

Valid on z/OS, z/VSE, and z/VM.

Use the MAXLEN keyword to define the maximum permission length for the user resource class being created.

The command has the following format:

```
TSS ADDTO(RDT) RESCLASS(resource name)
               RESCODE(hex code)
               MAXLEN(maxpermit)
```

maxpermit

A decimal value between 1 and 255 (rescode 01-3F) or 2 and 255 (rescode 101-13F).

This keyword is used with:

- The ADDTO command
- The FDT and RDT records
- MISC1(RDT) authority

Example: MAXLEN keyword

This example defines an RDT field length of 17:

```
TSS ADDTO(RDT) RESCLASS(#PRODUCT)
               RESCODE(03F)
               MAXLEN(17)
```

MAXTKTLF Keyword—Specify Maximum Ticket Life

Valid on z/OS.

Use the MAXTKTLF keyword:

- To specify the maximum ticket life in the KERBDFLT REALM record in the SDT
- When defining local principals in the security record of the user

When used in the local realm, this keyword has the following format:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME('kerberos-realm-name')
               MINTKTLF(max-ticket-life)
               MAXTKTLF(max-ticket-life)
               DEFTKTLF(default-ticket-life)
               KERBPASS(kerberos-password)
               CHKADDRS
```

When used in local principal, this keyword has the following format:

```
TSS ADDTO(acid) KERBNAME('kerberos-principal-name')
                MAXTKTLF(max-ticket-life)
```

max-ticket-life

Specifies the maximum ticket life in seconds.

When this keyword is used to define the maximum ticket life of the local realm, the DEFTKTLF and MINTKTLF keywords must also be specified. If only one of these keywords is specified, an error message is displayed. For additional information, see DEFTKTLF and MINTKTLF keywords.

When used to define the maximum ticket life of the local principle, this entry overrides the entry in the REALM MAXTKTLF record. DEFTKTLF and MINTKTLF keywords are not required for local principals.

This keyword is not used for foreign realms.

Limits: 1 - 2,147,483,647. Zero is not valid.

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- MISC3(SDT) authority to ADDTO or REMOVE records or fields in the SDT
- MISC8(LISTSDT) authority to list MAXTKTLF in the SDT

Examples: MAXTKTLF keyword

This example indicates a maximum ticket life of 24 hours in the default REALM SDT record:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME('LOCAL.CA.COM')
               MINTKTLF(30)
               MAXTKTLF(86400)
               DEFTKTLF(36000)
               KERBPASS(CHILDREN)
               CHKADDRS
```

This example changes the maximum ticket life:

```
TSS REPLACE(SDT) REALM(KERBDFLT)
                 MAXTKTLF(94000)
                 DEFTKTLF(36000)
                 MINTKTLF(30)
                 CHKADDRS
```

MCSALTG Keyword—Alternate Recovery Group

Valid on z/OS.

Use the MCSALTG keyword to assign an alternate group used in recovery. The IBM equivalent to this field is ALTG.

This keyword has the following format:

TSS ADDTO(*acid*) MCSALTG(*groupname*)

Capacity of list

One group per ACID

groupname length

One to eight characters

This keyword is used with:

- The commands REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Important! All MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

Examples: MCSALTG keyword

This example adds an alternate group name of X to ACID USERA:

```
TSS ADDTO(USERA) MCSALTG(X)
```

This example removes an alternate group name of X from USERA:

```
TSS REMOVE(USERA) MCSALTG(X)
```

MCSAUTH Keyword—Authorize Commands

Valid on z/OS.

Use the MCSAUTH keyword to authorize the operator commands that can be entered from the console. The IBM equivalent to this field is AUTH.

This keyword has the following format:

```
TSS ADDTO(acid) MCSAUTH(INFO|MASTER|SYS|IO|CONS|ALL)
```

Capacity of list

One authorization per ACID

INFO

Specifies that any informational commands can be entered from this console.

MASTER

Specifies that this is the master console.

SYS

Specifies that system control commands and informational commands can be entered from this console.

IO

Specifies that I/O control commands and informational commands can be entered from this console.

CONS

Specifies that console control commands and informational commands can be entered from this console.

ALL

Specifies that information, system control, I/O control, and console control commands can be entered from this console.

This keyword is used with:

- The commands REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN)authority

Important! All MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

Examples: MCSAUTH keyword

This example adds an authorization for all operator commands to ACID USERA:

```
TSS ADDTO(USERA) MCSAUTH(ALL)
```

This example removes an authorization for system commands:

```
TSS REMOVE(USERA) MCSAUTH(SYS)
```

MCSAUTO Keyword—Assign AUTO Keyword

Valid on z/OS.

Use the MCSAUTO keyword to specify whether the AUTO keyword is assigned to this console. The IBM equivalent to this field is AUTO.

This keyword has the following format:

TSS ADDTO(*acid*) MCSAUTO(YES|NO)

Capacity of list

One keyword per ACID

All MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Examples: MCSAUTO keyword

This example assigns the AUTO keyword to ACID USERA:

```
TSS ADDTO(USERA) MCSAUTO(YES)
```

This example removes the AUTO keyword of from USERA:

```
TSS REMOVE(USERA) MCSAUTO
```

MCSCMDS Keyword—Specify System

Valid on z/OS.

Use the MCSCMDS keyword to specify the system to which commands issued from this console are sent. The IBM equivalent to this field is CMDSYS.

This keyword has the following format:

```
TSS ADDTO(acid) MCSCMDS(*|sysname)
```

Capacity of list

One sysname per ACID

Keyword length

One to eight characters

*****

Specifies that commands are processed on the local system where the console is attached.

sysname

The name of the remote system to which the commands are sent.

Note: It is important to remember that all MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands REPLACE, ADDTO, REMOVE, and WHOHAS
- The MCSCMDS The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN)authority

Example: MCSCMDS keyword

This example sends commands to XYZ system:

```
TSS ADDTO(USERA) MCSCMDS(XYZ)
```

MCSDOM Keyword—Specify Delete Messages Received

Valid on z/OS.

Use the MCSDOM keyword to specify which delete operator messages (DOM) this console is to receive. The IBM equivalent to this field is DOM.

This keyword has the following format:

```
TSS ADDTO(acid) MCSDOM(NORMAL|ALL|NONE)
```

Capacity of list

One keyword per ACID

NORMAL

Receive all appropriate DOM requests.

ALL

Receive all DOM requests from the sysplex.

NONE

Do not receive any DOM requests.

MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands ADDTO, REPLACE, REMOVE, and WHOHAS
- The ACID types User DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Examples: MCSDOM keyword

This example receives all DOM requests:

```
TSS ADDTO(acid) MCSDOM(ALL)
```

This example receives no DOM requests:

```
TSS REMOVE(acid) MCSDOM(NONE)
```

MCSHC Keyword—Receive Hardcopy Message Set

Valid on z/OS.

Use the MCSHC keyword to specify if the extended console receives the hardcopy message set.

MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) MCSHC(YES|NO)
```

Capacity of list

One keyword per ACID

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, REPLACE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Example: MCSHC keyword

This example specifies that command responses are logged:

```
TSS ADDTO(acid) MCSHC(YES)
```

MCSINTIDS Keyword—Receive Messages for Console ID=0

Valid on z/OS.

Use the MCSINTIDS keyword to specify that the console receives messages directed to console ID zero.

MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) MCSINTDS(YES|NO)
```

Capacity of list

One keyword per ACID

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, REPLACE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Example: MCSINTIDS keyword

This example specifies that command responses are logged:

```
TSS ADDTO(acid) MCSINTIDS(YES)
```

MCSKEY Keyword—Assign Key Keyword

Valid on z/OS

Use the MCSKEY keyword to assign a KEY keyword to this console. The IBM equivalent to this field is KEY.

This keyword has the following format:

```
TSS ADDTO(acid) MCSKEY(keyword)
```

Capacity of list

One keyword per ACID.

keyword length

One to eight characters.

MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) authority as well as ACID(MAINTAIN) authority

Examples: MCSKEY keyword

This example adds an KEY keyword of X:

```
TSS ADDTO(acid) MCSKEY(X)
```

This example removes an KEY keyword of X:

```
TSS REMOVE(acid) MCSKEY(X)
```

MCSLEVL Keyword—Specify Messages Received

Valid on z/OS.

Use the MCSLEVL keyword to specify the messages received by this console. The IBM equivalent to this field is LEVEL.

This keyword has the following format:

`TSS ADDTO(`*acid*`) MCSLEVL(ALL|NB|R|I|CE|E|IN)`

Capacity of list

One keyword per ACID

ALL

Indicates that the console is to receive all messages.

NB

This console receives no broadcast messages.

R

This console receives the messages that require an operator reply.

I

This console receives immediate action messages.

CE

This console receives critical event action messages.

E

This console receives event action messages.

IN

This console receives informational messages.

MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Example: MCSLEVL keyword

This example specifies a console to receive informational messages:

```
TSS ADDTO(acid) MCSLEVL(IN)
```

MCSLOGC Keyword—Specify Hard Copy Log

Valid on z/OS.

Use the MCSLOGC keyword to specify whether command responses are logged in the hard copy log. The IBM equivalent to this field is LOGCMDRESP.

This keyword has the following format:

```
TSS ADDTO(acid) MCSLOGC(YES|NO)
```

Capacity of list

One keyword per ACID

MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Example: MCLOGC keyword

This example specifies that command responses should be logged:

```
TSS ADDTO(acid) MCSLOGC(YES)
```

MCSMFRM Keyword—Console Message Format

Valid on z/OS.

Use the MCSMFRM keyword to specify the display format for console messages. The IBM equivalent to this field is MFORM.

This keyword has the following format:

```
TSS ADDTO(acid) MCSMFRM(M|J|S|T|X)
```

Capacity of list

One keyword per ACID

M

Indicates that the system is to display the message ID only.

J

Indicates that the display is to include the job ID or name.

S

Indicates that the display is to include the name of the system originating the message.

T

Indicates that the display is to include a time stamp.

X

Indicates that messages that are exempt from jobname and system name formatting are ignored.

MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Examples: MCSMFRM keyword

This example displays message text only:

```
TSS ADDTO(acid) MCSMFRM(M)
```

This example removes the display of message text:

```
TSS REMOVE(acid) MCSMFRM(M)
```

MCSMGID Keyword—Migration ID

Valid on z/OS.

Use the MCSMGID keyword to specify whether a one-byte migration ID is assigned to this console. The IBM equivalent to this field is MIGID

This keyword has the following format:

```
TSS ADDTO(acid) MCSMGID(YES|NO)
```

Capacity of list

One keyword per ACID

MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Examples: MCSMGID keyword

This example adds a migration id to ACID USERA:

```
TSS ADDTO(USERA) MCSMGID(YES)
```

MCSMON Keyword—System Event Monitoring

Valid on z/OS.

Use the MCSMON keyword to specify how selected system events are monitored. The IBM equivalent to this field is MONITOR.

This keyword has the following format:

```
TSS ADDTO(acid) MCSMON(JOBNAMES|JOBNAMES-T|SESS|SESS-T|STATUS)
```

Capacity of list

One keyword per ACID

JOBNAMES

Specifies that each jobname is displayed when the job starts and ends. Unit record allocation is displayed when the job starts. If a job abends, the jobname will appear in a diagnostic message.

JOBNAMES-T

Specifies that the time is displayed with the jobname. The time appears in hh:mm:ss. format.

SESS

Specifies that the user identifier for each time—sharing terminal is displayed when the session starts and ends. If a session abends, the user identifier will appear in a diagnostic message.

SESS-T

Specifies that the time is displayed with the user identifier. The time appears in hh:mm:ss format.

STATUS

Specifies that the data set names and volume serial numbers of data sets with dispositions of KEEP, CATG, and UNCATLG are displayed whenever they are freed.

All MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Example: MCSMON keyword

This example specifies the display of jobs and the time they begin and end:

```
TSS ADDTO(acid) MCSMON(JOBNAMES)
```

MCSROUT Keyword—Console Routing Codes

Valid on z/OS.

Use the MCSROUT keyword to specify the routing codes assigned to the console. The IBM equivalent to this field is ROUTCODE.

This keyword has the following format:

```
TSS ADDTO(acid) MCSROUT(NONE|ALL|nnn,...)
```

Valid values

1 through 128.

ALL

Specifies all routing codes, 1 through 128.

nnn

Specific routing codes assigned to the console.

NONE

Console will receive no routing codes, except the master console which is always 1 and 2.

Default: NONE

MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands ADDTO, REPLACE, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Example: MCSROUT keyword

This example specifies all routing codes:

```
TSS ADDTO(acid) MCSROUT(ALL)
```

MCSSTOR Keyword—Message Queue Storage

Valid on z/OS.

Use the MCSSTOR keyword to specify the amount of storage, in megabytes, used for message queuing. The IBM equivalent to this field is STORAGE.

This keyword has the following format:

```
TSS ADDTO(acid) MCSSTOR(nnnn)
```

Capacity of list

One keyword per ACID

Valid values

One to 2000.

MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Example: MCSSTOR keyword

This example defines message queuing storage to 2000 megabytes:

```
TSS ADDTO(acid) MCSSTOR(2000)
```

MCSUD Keyword—Specify Messages Received

Valid on z/OS.

Use the MCSUD keyword to specify whether this console is to receive undelivered action messages and WTOR messages. The IBM equivalent to this field is OPERUD.

This keyword has the following format:

```
TSS ADDTO(acid) MCSUD(YES|NO)
```

Capacity of list

One keyword per ACID

Note: It is important to remember that all MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Example: MCSUD keyword

This example specifies that this console is to receive undelivered messages:

```
TSS ADDTO(acid) MCSUD(YES)
```

MCSUNKNIDS Keyword—Receive Messages for Unknown Console IDs

Valid on z/OS.

Use the MCSUNKNIDS keyword to specify whether the console receives messages directed to unknown console IDs, such as one-byte IDs.

MCS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) MCSUNKNIDS(YES|NO)
```

Capacity of list

One keyword per ACID

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, REPLACE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC8(MCS) and ACID(MAINTAIN) authority

Example: MCSUNKNIDS keyword

This example specifies that command responses are logged:

```
TSS ADDTO(acid) MCSUNKNIDS(YES)
```

MEMLIMIT Keyword—Maximum Memory Space

Valid on z/OS.

Use the MEMLIMIT keyword to specify the maximum number of bytes of non-shared memory space that this user can allocate.

This keyword has the following format:

```
TSS ADD(TESTID) MEMLIMIT(value)
```

The MEMLIMIT value can be from 0 to 16,777,215 followed by a letter indicating a multiplier used to calculate the total number of bytes (M for megabyte, G for gigabyte, T for terabyte, and P for petabyte). The maximum petabyte value is 16383P. The maximum terabyte value is 16776192T. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. You can remove this field from the record by changing it to a null value (change user MEMLIMIT()). MEMLIMIT is only valid at z/OS 1.6 and above.

The following table describes the multiplier value used to calculate the total number of bytes:

Multiplier	Decimal	Binary	Hex
M = Megabyte	1,048,576	2**20	00000000 00100000
G = Gigabyte	1,073,741,824	2**30	00000000 40000000
T = Terabyte	1,099,511,627,776	2**40	00000100 00000000
P = Petabyte	1,125,899,906,842,624	2**50	00040000 00000000

This keyword is used with:

- The commands CREATE, ADD, ADDTO, REMOVE, and REPLACE
- The ACID type USER
- ACID(MAINTAIN) or ACID(CREATE) authority

Examples: MEMLIMIT keyword

This example assigns a user a non-shared memory limit of 12345 megabytes:

```
TSS ADD(TESTID) MEMLIMIT(12345M)
```

This example removes a non-shared memory limit from a user:

```
TSS REMOVE(TESTID) MEMLIMIT
```

MINTKTLF Keyword—Minimum Ticket Life

Valid on z/OS.

Use the MINTKTLF keyword to specify the minimum ticket life in the KERBDFLT REALM record in the SDT.

This keyword has the following format:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME('kerberos-realm-name')
               MINTKTLF(min-ticket-life)
               MAXTKTLF(max-ticket-life)
               DEFTKTLF(default-ticket-life)
               KERBPASS(kerberos-password)
               CHKADDRS
```

min-ticket-life

Specifies the minimum ticket life in seconds.

This keyword is only applicable when defining the KERBDFLT realm record (not foreign realms). If MINTKTLF is specified, the DEFTKTLF and MAXTKTLF keywords must also be specified. If only one of these keywords is specified, am error message is displayed. For additional information, see the DEFTKTLF and MAXTKTLF keywords.

Limits: 1 - 2,147,483,647. Zero is not valid.

This keyword is used with:

- The commands ADDTO, DELETE, REMOVE, REPLACE, and LIST
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- MISC3(SDT) authority to ADDTO or REMOVE records or fields in the SDT
- MISC8(LISTSDT) authority to list MINTKTLF in the SDT

Example: MINTKTLF keyword

This example indicates a minimum ticket life of 30 seconds in the default REALM SDT record:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME('LOCAL.CA.COM')
               MINTKTLF(30)
               MAXTKTLF(86400)
               DEFTKTLF(36000)
               KERBPASS(CHILDREN)
               CHKADDRS
```

MISC1 Keyword—Authority to Administer Functions

Valid on z/OS, z/VSE, and z/VM.

Use the MISC1 keyword to give or to remove an CA Top Secret administrator's authority to perform one or more administrative functions.

This keyword has the following format:

```
TSS ADMIN(acid) MISC1(authority level(s))
```

This keyword is used with:

- The commands ADMIN, DEADMIN, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA

MISC1 Authority Level

The CA Top Secret administrator may specify any or all of the following MISC1 authorities:

LCF

Authorizes administrators to assign LCF command and transaction restrictions ((X)COMMAND and (X)TRANSACTION) to ACIDs

INSTDATA

Authorizes administrators to associate installation data with ACIDs It also authorizes administrators to ADD Dynamic Update Facility attributes, DUFXTR and DUFUPD, to ACIDs

USER

Authorizes administrators to administer the use of unownable installation-defined resources (USERx).

LTIME

Authorizes administrators to set time intervals that determine when CA Top Secret will lock an unused terminal for ACIDs

SUSPEND

Authorizes administrators to administer the SUSPEND attribute to ACIDs

NOATS

Authorizes administrators to prevent ACIDs within their scope (in CICS, IMS, and CA-IDMS) from signing on via ATS (Automatic Terminal Signon).

RDT

Authorizes the administrator to maintain and list the RDT Record (Resource Descriptor Table) and the FDT (Field Descriptor Table) Record.

TSSSIM

Authorizes administrators to use the TSSSIM utility.

ALL

Authorizes administrators to use all of the authorities.

Examples: MISC1 keyword

This example allows the administrator to suspend and unsuspend ACIDs within his scope:

```
TSS ADMIN(AMCDCA) MISC1(SUSPEND)
```

This example removes all lower level administrative authorities from AMCDCA:

```
TSS DEADMIN(AMCDCA) MISC1(ALL)
```

MISC2 Keyword—Authority to Administer Functions

Valid on z/OS, z/VSE, and z/VM.

Use the MISC2 keyword to give, or to remove, an CA Top Secret administrator's authority to perform one or more administrative functions.

This keyword has the following format:

```
TSS ADMIN(acid) MISC2(authority level(s))
```

This keyword is used with:

- The commands ADMIN, DEADMIN, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA

MISC2 Authority Levels

The CA Top Secret administrator can specify any or all of the following MISC2 authorities:

ALL

Authorizes administrators to use all of the authorities.

SMS

Authorizes an administrator to ADD, REMOVE, and REPLACE the following SMS fields to an ACID within his scope: SMSAPPL, SMSDATA, SMSMGMT, and SMSSTOR.

TSO

Authorizes an administrator to ADD, REMOVE, and REPLACE the following TSO UADS fields for an ACID within his scope: TSOCOMMAND, TSODEST, TSODEFPRFG, TSOSCLASS, TSOHCLASS, TSOJCLASS, TSOLACCT, TSOLPROC, TSOLSIZE, TSOMCLASS, TSOMSIZE, TSOOPT, TSOUDATA, and TSOUNIT.

NDT

Authorizes the administrator to maintain data in the Node Descriptor Table (NDT).

DLF

Gives the ability to ADD and REMOVE data sets from the DLF (Data Lookaside Facility) Record.

APPCLU

Authorizes the administrator to maintain the APPCLU Record.

WORKATTR

Authorizes the administrator to ADD and REMOVE the following sysout delivery and account number information: WAACCNT, WAADDR1, WAADDR2, WAADDR3, WAADDR4, WABLDG, WADEPT, WANAME, and WAROOM.

TARGET

Gives the ability to:

Administer nodes associated with ACIDs

Perform commands where TARGET is specified as part of the command

Examples: MISC2 keyword

This example authorizes an administrator to set SMS data fields for users within his scope:

```
TSS ADMIN(acid) MISC2(SMS)
```

This example removes MISC2 authority:

```
TSS DEADMIN(acid) MISC2(ALL)
```

MISC3 Keyword—Authority to Administer Functions

Valid on z/OS, z/VSE, and z/VM.

Use the MISC3 keyword to give, or to remove, an CA Top Secret administrator's authority to perform one or more additional administrative functions.

This keyword has the following format:

```
TSS ADMIN(acid) MISC3(authority level(s))
```

This keyword can be used with:

- The commands ADMIN, DEADMIN, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA

MISC3 Authority Levels

The CA Top Secret administrator may specify the following MISC3 authorities:

ALL

Authorizes administrators to use all of the authorities.

SDT

Authorizes an administrator to maintain and list the SDT (Static Data Table) record.

PTOK

(Valid on z/OS only.) Authorizes an administrator to issue the TSS P11TOKEN command.

Examples: MISC3 keyword

This example authorizes an administrator to define record elements to the SDT and PERMIT them to users within his scope:

```
TSS ADMIN(acid) MISC3(SDT)
```

This example removes MISC3 authority:

```
TSS DEADMIN(acid) MISC3(SDT)
```

MISC4 Keyword—Authority to Administer Functions

Valid on z/OS, z/VSE, and z/VM.

Use the MISC4 keyword to give, or to remove, an CA Top Secret administrator's authority to perform one or more additional administrative functions.

This keyword has the following format:

```
TSS ADMIN(acid) MISC4(authority level(s))
```

This keyword can be used with:

- The commands ADMIN, DEADMIN, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA

MISC4 Authority Levels

The CA Top Secret administrator may specify the following MISC4 authorities:

ALL

Authorizes administrators to use all of the authorities.

CERTUSER

Allows the security administrator to maintain user ACIDs.

CERTAUTH

Allows the security administrator to maintain certificate authority ACIDs.

CERTSITE

Allows the security administrator to maintain site certificates ACIDs.

CERTLIST

Allows the security administrator to list digital certificate information.

CERTGEN

Allows the security administrator to generate digital certificates.

CERTEXPO

Allows the security administrator to export digital certificates.

CERTCHEK

Allows the security administrator to display information about digital certificates.

KERBUSER

Allows the security administrator to map foreign principal names to individual user IDs.

Examples: MISC4 keyword

This example authorizes an administrator to export digital certificates for users within the administrator's scope:

```
TSS ADMIN(USER01) MISC4(CERTEXPO)
```

This example removes MISC4 authority:

```
TSS DEADMIN(USER01) MISC4(CERTEXPO)
```

MISC5 Keyword—Authority to Administer Functions

Valid on z/OS and z/VM.

Use the MISC5 keyword to give or to remove an CA Top Secret administrator's authority to perform one or more administrative functions. This keyword is part the DCLAS tracking record that helps determine which data (files, data sets, resources) pertains to which regulation. This record resides in the VSAM file.

This keyword has the following format:

```
TSS ADMIN(acid) MISC5(authority level(s))
```

This keyword can be used with:

- The commands ADMIN, DEADMIN, and WHOHAS
- The ACID types MSCA and SCA

MISC5 Authority Levels

The CA Top Secret administrator may specify the following MISC5 authority:

ALL

Authorizes administrators to use all of the authorities.

DCLADMIN

Allows the security administrator to maintain the Data Classification (DATACLAS) record.

DCLLIST

Allows the security administrator to list the Data Classification (DATACLAS) record.

MLSADMIN

Allows the security administrator to maintain and list the Multilevel Security (MLS) record.

SGVADMIN

Allows the security administrator to maintain the SIGVER record.

SGVLIST

Allows the security administrator to list the SIGVER record.

Examples: MISC5 keyword

This example authorizes an administrator to define security labels for MLS security:

```
TSS ADMIN(techsca) MISC5(mlsadmin)
```

This example removes MISC5 authority:

```
TSS DEADMIN(techsca) MISC5(mlsadmin)
```

MISC7 Keyword—Authority to Administer Functions

Valid on z/OS.

Use the MISC7 keyword to give or remove an CA Top Secret administrator's authority to perform one or more additional administrative functions.

This keyword has the following format:

```
TSS ADMIN(acid) MISC7(authority level(s))
```

This keyword can be used with:

- The commands ADMIN, DEADMIN, and WHOHAS
- The Acid types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA

MISC7 Authority Levels

The CA Top Secret administrator may specify the following MISC7 authorities:

RSTDACC

Authorizes an administrator to associate the RSTDACC attribute with ACIDs within his scope.

MISC8 Keyword—Authority to Administer Functions

Valid on z/OS, z/VSE, and z/VM.

Use the MISC8 keyword to give, or to remove, an CA Top Secret administrator's authority to list the contents of the RDT, FDT or STC or to use the ASUSPEND administrative function.

This keyword has the following format:

```
TSS ADMIN(acid) MISC8(authority level(s))
```

This keyword is used with:

- The commands ADMIN, DEADMIN, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA

MISC8 Authority Levels

The CA Top Secret administrator may specify any or all of the following MISC8 authorities:

LISTRDT

Authorizes an administrator to list the RDT and FDT Records but not to change them. MISC1(RDT) authority is required to maintain the RDT and FDT Records.

LISTSTC

Authorizes the administrator to list the contents of the Started Task Table but not to change it. MISC9(STC) authority is required to define started tasks to the Started Task Table.

LISTAPLU

Authorizes the administrator to list the contents of the APPCLU Record but not to change it. MISC2(APPCLU) authority is required to maintain the APPCLU record.

LISTSDT

Authorizes the administrator to list the contents of the Static Data Table (SDT) but not to change it. MISC3(SDT) authority is required to maintain the SDT records.

MCS

Authorizes the administrator to issue the Multiple Console Support (MCS) commands for ACIDs.

NOMVSDF

Allows the security administrator to provide an acid attribute (NOOMVSDF) which will prevent a user that does not have a UID/GID from inheriting the OMVS default user and group.

PWMAINT

Authorizes the administrator to do password maintenance on acids with their scope. This will allow the use of the PASSWORD keyword on any command, or the SUSPEND keyword on the REMOVE command, without specifying ACID(MAINTAIN) or MISC1(SUSPEND).

REMASUSP

Authorizes the administrator to issue the TSS REMOVE(acid) ASUSPEND command for ACIDs. In addition to MISC8(REMASUSP), the administrator must also have MISC1(SUSPEND) or MISC8(PWMAINT) authority to remove the ASUSPEND attribute.

ALL

Authorizes the administrator to use all of the authorities.

Examples: MISC8 keyword

This example gives the administrator the authority to remove an administrative suspension from an ACID within his scope:

```
TSS ADMIN(TECHSCA) MISC8(REMASUSP)
```

This example gives TECHVCA the authority to list the RDT Record:

```
TSS ADMIN(TECHVCA) MISC8(LISTRDT)
```

This example removes all MISC8 authority:

```
TSS DEADMIN(acid) MISC8(ALL)
```

MISC9 Keyword—Authority to Administer Functions

Valid on z/OS, z/VSE, and z/VM.

Use the MISC9 keyword to give, or to remove, a TSS administrator's authority to perform one or more high-level administrative functions.

This keyword has the following format:

```
TSS ADMIN(acid) MISC9(authority level(s))
```

This keyword is used with:

- The commands ADMIN, DEADMIN, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA

MISC9 Authority Levels

The CA Top Secret administrator may specify any or all of the following MISC9 authorities:

BYPASS

Authorizes an administrator to associate the following bypass attributes with ACIDs within his scope: NODSNCHK, NOVOLCHK, NORESCHK, NOVMDCHK, NOLCFCHK, and NOSUBCHK.

TRACE

Authorizes the administrator to associate the TRACE attribute with ACIDs within his scope.

CONSOLE

Authorizes the administrator to administer the CONSOLE attribute.

MASTFAC

Authorizes the administrator to associate a region control ACID with a multiuser address space facility.

MODE

Authorizes the administrator to associate any CA Top Secret owned security mode with an ACID. Only MSCA and SCA type acids can use the MODE keyword in a TSS PERMIT(acid) command.

STC

Authorizes 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.

GLOBAL

Authorizes the administrator to use LCF and administrative authorities for all users via the ALL record.

GENERIC

Authorizes the administrator to use the WHOOWNS function to obtain a list of all resources owned within his administrative scope. RESOURCE(INFO) authority only allows an ACID to obtain data on specific resources.

ALL

Authorizes the administrator to use all of the authorities.

Examples: MISC9 keyword

This example authorizes an administrator to set security modes for users within his scope:

```
TSS ADMIN(TECHSCA) MISC9(MODE)
```

This example authorizes CA Top Secret administrator TECHVCA to assign the NOSUBCHK attribute to job submission ACIDs:

```
TSS ADMIN(TECHVCA) MISC9(BYPASS)
```

This example gives an administrator in the Software Development Department the authority to define region control ACIDs for CICS:

```
TSS ADMIN(DEVDCA) MISC9(MASTFAC,STC)
```

This example removes MISC9 authority:

```
TSS DEADMIN(acid) MISC9(ALL)
```

MMAPAREA Keyword—Maximum Data Space Pages for HFS Mappings

Valid on z/OS.

Use the MMAPAREA keyword to specify the maximum amount of dataspace storage (pages) that can be allocated for memory mapping of HFS. This field overrides the MAXMMAPAREA parameter in the BPXPRMxx member of PARMLIB for this user

MMAPAREA is equivalent to MMAPAREAMAX in RACF.

This keyword has the following format:

```
TSS ADD(acidname) MMAPAREA(nnnnnnn)
```

nnnnnnn

The maximum amount of dataspace storage that can be allocated for HFS memory mapping.

Range: 1 to 16,777,216

Default: USS takes the system defaults set in BPXPRMxx.

This keyword is used with the ADDTO, REPLACE, REMOVE, and WHOHAS commands.

Examples: MMAPAREA keyword

This example assigns an acid a value of 10 for MMAPAREA:

```
TSS ADD(TESTID) MMAPAREA(10)
```

This example removes MMAPAREA from the acid:

```
TSS REMOVE(TESTID) MMAPAREA
```

MODE Keyword—Operating Mode

Valid on z/OS, z/VSE, and z/VM.

Use the MODE keyword to:

- Assign ownership of CA Top Secret operating modes to the master SCA (MSCA)
- Specify an operating MODE for a user, control or profile ACID.

A user or profile MODE will override the global and facility MODES.

When used with ADDTO, this keyword has the following format:

```
TSS ADDTO(sca acid) MODE(DORM,WARN,IMPL,FAIL)
```

When used with PERMIT this keyword has the following format:

```
TSS PERMIT(USER01) MODE(WARN)
                   FACILITY(BATCH,TSO)
```

Generally, administrators should only upgrade the mode.

This keyword is used with:

- The commands ADDTO, REMOVE, WHOOWNS, WHOHAS, PERMIT, and REVOKE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MSCA ownership of all modes before they can be permitted. Modes can only be administered by an SCA ACID with MISC9(MODE) authority, via the TSS ADMIN function

MODE Keyword—Specify Operating MODE

Valid on z/OS and z/VM.

Use the MODE keyword to specify an operating MODE for a user, control or profile ACID.

This keyword has the following format:

```
TSS PERMIT(acid|profile) MODE(DORM|WARN|IMPL|FAIL)
```

Examples: MODE keyword

This example forces a user to run in IMPL mode:

```
TSS PERMIT(PAYUSER) MODE(IMPL)
```

This example changes this specification to FAIL mode:

```
TSS REVOKE(PAYUSER) MODE(IMPL)

TSS PERMIT(PAYUSER) MODE(FAIL)
```

MRO Keyword—Maintain Security Records

Valid on z/OS.

Use the MRO keyword to store or remove user and profile Security Records in Common System Storage (CSA or ECSA) so that security information is accessible to all online regions.

Note: The CICS MRO Optimized Signon (MOS) feature (that the MRO command previously was used to activate) has been removed from CICS.

This keyword has the following format:

```
TSS ADDTO(region acid) MRO
```

MRO must be associated with the master region ACID, not with individual user ACIDs.

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The User type ACID only
- ACID (MAINTAIN) authority

Examples: MRO keyword

This example activates MRO for a specific online region (ONLINE1):

```
TSS ADDTO(ONLINE1) MRO
```

This example removes the MRO attribute:

```
TSS REMOVE(ONLINE1) MRO
```

MULTIPW Keyword—Maintain Multiple Password

Valid on z/OS and z/VM.

Use the MULTIPW keyword to assign or remove multiple password attributes, which means ACIDs need a different password to access each facility.

This keyword has the following format:

```
TSS ADD(acid) FAC(facility)
              PASSWORD(pswd[,[interval][,EXP]])
              MULTIPW
```

Note the following:

- All keyword operands associated with MULTIPW are required.
- The following PASSWORD operands are positional:
- *pswd* (required)
- *interval* (optional)
- EXP (optional)
- See PASSWORD for the ADDTO/REMOVE command for complete information.
- When a MULTIPW password has been set, the general password set for the ACID becomes associated with FACILITY(*ALL*). However, the syntax for manipulating the general password remains:
  `TSS ADD|REMOVE|REPLACE(acid) PASSWORD(...)`

This keyword is used with:

- The commands CREATE, ADDTO, REPLACE, and REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority
- FAC(*xxx*), where *xxx* is the facility being added

Examples: MULTIPW keyword

This example indicates that USER99 may have a different password to access each facility:

```
TSS ADDTO(USER99) PASSWORD(TAQRAC)
                  FACILITY(TSO)
                  MULTIPW
```

This example allows the following entries for USER99:

```
TSS ADDTO(USER99) PASSWORD(BUZWRD)
                  FACILITY(IMS)
                  MULTIPW

TSS ADDTO(USER99) PASSWORD(SUPAV1)
                  FACILITY(CICS)
                  MULTIPW
```

This example removes the MULTIPW attribute removes all facility specific passwords from the user's security record:

```
TSS REMOVE(USER99) PASSWORD(SUPAV1)
                   FACILITY(CICS)
                   MULTIPW
```

To reset a password from a facility-specific password back to the user's global password, remove the facility associated with the password:

```
TSS REMOVE(USER99) FAC(idmsprod)
```

This removes access to the facility along with the facility-specific password, re-add the facility to preserve user access to the facility.

```
TSS REMOVE(USER99) FAC(idmsprod)
```

NADATE and NATIME Keywords—Date Certificate Expires

Valid on z/OS.

Use the NADATE and NATIME keywords to specify the effective date and time that the digital certificate expires. If no date is specified, it defaults to the active day and time plus 1 year. The expire date (NADATE) specified must be later than the active date (NBDATE).

The year specified must fall within the range 1950-2049. If an expire date is not also specified, the NBDATE year specified must fall within the range 1950-2048, since the NADATE date defaults to the active day and time plus one year.

The certificate DATE FORMAT are not governed by the DATE parm in CA Top Secret. The format is MM/DD/YY.

The NADATE specifies the “not active after” date of a certificate.

The NATIME specifies the “not active after” time of a certificate.

This keyword has the following format:

```
TSS REKEY(acid) NADATE(mm/dd/yy)
                NATIME(hh:mm:ss)
```

Date and time fields are optional, except if time is specified, date is required.

This keyword is used with:

- The GENCERT, REKEY and RENEW commands
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority
- MISC4(CERTSITE) authority for CERTSITE ACID
- MISC4(CERTAUTH) for CERTAUTH ACID

Example: NADATE and NATIME keywords

This example specifies an effective date:

```
TSS REKEY(user1) DIGICERT(cert0001)
                 NEWDIGIC(CERT0002)
                 NADATE(09/01/07)
                 NATIME(00:00:01)
```

This example specifies an effective date:

```
TSS GENCERT(user1) DIGICERT(cert0001)
                   DCDSN(user1.cert.data)
                   NADATE(09/01/02)
                   NATIME(00:00:01)
```

NAME Keyword—Associate ACID with a Name

Valid on z/OS, z/VSE, and z/VM.

Use the NAME keyword to associate the ACID with a name for further identification.

The NAME field is provided for reporting and documentation purposes. CA Top Secret displays the NAME field as part of message TSS7000I when a user signs on to a facility. NAME is also displayed in the TSSUTIL report for initiation or signon records.

This keyword has the following format:

TSS CREATE(*acid*) NAME(['])*character name*['']) ...

The NAME keyword may contain from one to 32 characters.

The entire name must be delimited by single quotes if it contains blanks, lower case, or special characters.

If the name contains an apostrophe ('), use two single quotes within the name; for example: NAME('Patrick O''Neil').

If the name contains special characters such as '^' and '~', in order for them to display, set control option OPTIONS(73 or 16) and run TSS LIST via TSSCFILE or CA Top Secret Workstation Option. Without OPTIONS (73 or 16) set, the entire contents of the NAME field are automatically uppercased.

This keyword is used with:

- The commands ADDT, REPLACE, and CREATE
- The ACIDs User, Profile, DEPARTMENT, DIV, ZONE, DCA, VCA, ZCA, LSCA, and SCA

Example: NAME keyword

This example assigns a name to a division's ACID:

```
TSS CREATE(SHIPDIV) NAME('SHIPPING DIVISION')...
```

NBDATE and NBTIME Keywords—Date Certificate Activates

Valid on z/OS.

Use the NBDATE and NBTIME keywords to specify the effective date and time the digital certificate becomes active. If not time is specified, it defaults to 0000000. If no date is specified, it defaults to the current day and time. The active date (NBDATE) specified must be earlier than the expire date (NADATE)

The year specified must fall within the range 1950-2049. If an expire date is not also specified, the NBDATE year specified must fall within the range 1950—2048, since the NADATE date defaults to the active day and time plus one year.

The certificate DATE FORMAT are not governed by the DATE parm in CA Top Secret. The format is MM/DD/YY.

The NBDATE specifies the “not before” date of a certificate.

The NBTIME specifies the “not before” time of a certificate.

This keyword has the following format:

```
TSS REKEY NBDATE(mm/dd/yy)
          NBTIME(hh:mm:ss)
```

Date and time fields are optional, except if time is specified, date is required.

This keyword is used with:

- The GENCERT, REKEY and RENEW commands
- The ACID types User, DCA, VCA, ZCA LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN)authority
- MISC4(CERTSITE) for CERTSITE ACIDs
- MISC4(CERTAUTH) for CERTAUTH ACIDs

Example: NBDATE and NBTIME keywords

This example specifies a deactivation date:

```
TSS REKEY(user1) DIGICERT(cert0001)
                 NEWDIGIC(CERT0002)
                 NBDATE(09/01/07)
                 NBTIME(11:59:59)
```

This example specifies a deactivation date:

```
TSS GENCERT(user1) DCDSN(user1.cert.data)
                   DIGICERT(cert0001)
                   NBDATE(09/01/02)
                   NBTIME(11:59:59)
```

NEWDIGIC Keyword—Name Certificate

Valid on z/OS.

Use the NEWDIGIC keyword to specify the name that identifies a digital certificate within an acid record. The NEWDIGIC name must be entered as part of all ROLLOVER and REKEY functions since this keyword indicates the new name used in the digital certificate.

This keyword has the following format:

```
TSS REKEY NEWDIGIC(name)
```

name

Specifies a “case-sensitive” ID that identifies a NEW digital certificate with the user acid.

Range: 1 to 8 characters

The keyword is used with:

- The commands ROLLOVER and REKEY
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority
- MISC4(CERTSITE) for CERTSITE ACIDs
- MISC4(CERTAUTH) for CERTAUTH ACIDs

Example: DIGICERT keyword

This example generates a new (NEWDIGIC) based on certificate in DIGICERT field:

```
TSS ROLLOVER(myacid) DIGICERT(TEST)
                    NEWDIGIC(NEWTEST)
```

This example generates a new certificate (NEWDIGIC) based on certificate in DIGICERT field:

```
TSS REKEY(myacid) DIGICERT(TEST)
                  NEWDIGIC(NEWTEST)
```

NEWLABLC Keyword—Label Associated with Certificate

Valid on z/OS.

Use the NEWLABLC keyword to specify an optional and case-sensitive label associated with the certificate being added to the user. Spaces are allowed if you use single quotes. This label is used as a descriptive identifier for a certificate, and must be unique for the individual user. If a label is not specified, the label field will default to the value specified within the NEWDIGIC keyword.

This keyword has the following format:

```
TSS ADDTO NEWLABLC(label-name)
```

label-name

Specifies a label name for the new certificate. If a label is not specified, the label field will default to the value specified within the NEWDIGIC keyword.

Length: Up to 32 characters

This keyword is used with:

- The commands ADDTO, GENCERT, LIST, REKEY, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority
- MISC4(CERTSITE) authority for CERTSITE ACIDs
- MISC4(CERTAUTH) authority for CERTAUTH ACIDs

Example: NEWLABLC keyword

This example associates the label new_certificate_name with the certificate.

```
TSS REKEY(user1) DIGICERT(cert0001)
                 NEWDIGIC(CERT0002)
                 KEYSIZE(512)
                 NEWLABLC(new_certificate_name)
```

NISTECC Keyword—Generate Key with NISTECC

Valid on z/OS

Use the NISTECC keyword to generate a key pair using the National Institue of Standards and Technology (NIST) algorithm instead of the RSA algorithm. This parameter cannot be used with the ICSF, DSA, or BPECC parameters.

Because the key is generated using ICSF PKCS #11 functions, the ICSF subsystem must be operational and configured for PKCS #11 operation. Minimally, ICSF must be at the HCR7770 level. The private key is placed in the CA Top Secret database. When ICSF is at the HCR7780 level or higher, specify PCICC or PKDSLBLto store the private key in ICSF instead of the CA Top Secret database by.

This keyword has the following format:

```
TSS GENCERT(acid) DIGICERT(8-byte name) NISTECC
```

The keyword is used with:

- The GENCERT and REKEY commands
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority for users, MISC4(CERTSITE) for CERTSITE ACID, and MISC4(CERTAUTH) for CERTAUTH ACID

NOADSP Keyword—Prevent Data Set Security

Valid on z/OS.

Use the NOADSP keyword to:

- Prevent data sets, created by an ACID, from being automatically secured by z/OS by setting the RACF bit
- Define an ACID used to create data sets that cannot be automatically protected by CA Top Secret

This keyword has the following format:

```
TSS ADDTO(acid) NOADSP
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: NOADSP keyword

This example allows USER01 to create data sets that do not automatically have a RACF bit set by CA Top Secret:

```
TSS ADDTO(USER01) NOADSP
```

This example removes the NOADSP attribute:

```
TSS REMOVE(USER01) NOADSP
```

Note: Only applicable in non-always call environments.

NOATS Keyword—Fail Automatic Terminal Signons

Valid on z/OS and z/VSE.

Use the NOATS keyword to fail Automatic Terminal Signons. If a user attempts to execute a transaction in CICS IMS or CA-IDMS and has not signed on at an initial terminal session, CA Top Secret attempts to sign on the user to the application using an ACID identical to the local session terminal ID. This is done through Automatic Terminal Signon (ATS). If an ACID exists that matches the local session terminal ID but has the NOATS attribute, the ATS sign on of the last ACID fails. Depending on the options installed for the multi-user address space, a default user may be assigned.

Automatic Terminal Signon is intended to assign a non-powerful ACID to a terminal where temporary workers perform routine transaction processing without formal signons. The NOATS attribute is applied to a powerful ACID that might inadvertently be used for ATS processing. Such an ACID could be an CA Top Secret administrator or a privileged application client. Assigning NOATS to a privileged ACID is the administrator's responsibility and is not assigned by default.

This keyword has the following format:

```
TSS ADDTO(acid) NOATS
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC1(NOATS) authority

Note: An administrator already defined to CA Top Secret will, by default, not have the NOATS administrative authority.

Examples: NOATS keyword

This example prevents USER01 from signing on via ATS.

```
TSS ADDTO(USER01) NOATS
```

This example removes the NOATS attribute:

```
TSS REMOVE(USER01) NOATS
```

NODSNCHK Keyword—Bypasses Access Security Checks

Valid on z/OS, z/VSE, and z/VM.

Use the NODSNCHK keyword to specify that no data sets name checks are performed. CA Top Secret bypasses all data set access security checks. All data set access is audited.

NODSNCHK is intended to only be applied to special products, such as DASD space managers, which access many data sets. Avoid applying NODSNCHK to user ACIDs.

This keyword has the following format:

```
TSS ADDTO(acid) NODSNCHK
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC9(BYPASS) authority

Examples: NODSNCHK keyword

This example enables BYPACID to bypass security for data sets and thus access any data set at the installation:

```
TSS ADDTO(BYPACID) NODSNCHK
```

This example removes the NODSNCHK attribute:

```
TSS REMOVE(BYPACID) NODSNCHK
```

NOLCFCHK Keyword—Execute any Command

Valid on z/OS and z/VSE.

Use the NOLCFCHK keyword to allow an ACID to execute any command or transaction for all facilities, regardless of LCF (Limited Command Facility) restrictions. Auditing will occur. If the NOLCFCHK attribute is added to an ACID, then that ACID's terminal cannot be locked.

This keyword has the following format:

```
TSS ADDTO(acid) NOLCFCHK
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC9(BYPASS) authority

Examples: NOLCFCHK keyword

This example allows a user to execute all commands and programs:

```
TSS ADDTO(Q2004) NOLCFCHK
```

This example removes the NOLCFCHK attribute:

```
TSS REMOVE(Q2004) NOLCFCHK
```

NOPERMIT Keyword—Prevent Automatic Access

Valid on z/OS, z/VSE, and z/VM.

Use the NOPERMIT to prevent an owner from being automatically permitted access to a resource whose ownership was transferred via the ADDTO command function.

This keyword has the following format:

```
TSS ADDTO(acid) resource(prefix)
                UNDERCUT
                NOPERMIT
```

This keyword is used with:

- The commands CREATE and ADDTO
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- Resource(OWN) authority

Examples: NOPERMIT keyword

In this example, ownership of data set SYS.01 is transferred from user OLDOG to user NWDOG. After the transfer OLDOG still has to access the data set even though ownership was transferred:

```
TSS ADDTO(NWDOG) DSNAME(SYS.01)
                 UNDERCUT
```

This example transfers ownership of the the data set to NWDOG and prevents OLDOG from having permission to access the data set:

```
TSS ADDTO(NWDOG) DSNAME(SYS.01)
                 NOPERMIT
                 UNDERCUT
```

NOPWCHG Keyword—Prevent Password Changes

Valid on z/OS and z/VM.

Use the NOPWCHG keyword to prevent ACIDs from changing passwords at signon or initiation.

This keyword has the following format:

```
TSS ADDTO(acid) NOPWCHG
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(CREATE) authority
- Administrators who does not have ACID(CREATE) authority can add NOPWCHG to users within their scope if given UPDATE access to entity TSSCMD.USER.ADDTO.NOPWCHG in the CASECAUT resource class
- Administrators who do not have ACID(CREATE) authority can remove NOPWCHG from users within their scope if given UPDATE access to entity TSSCMD.USER.REMOVE.NOPWCHG in the CASECAUT resource class

Examples: NOPWCHG keyword

This example prevents a group of users from changing their passwords:

```
TSS ADDTO(GROUPA) NOPWCHG
```

This example removes the NOPWCHG attribute:

```
TSS REMOVE(GROUPA) NOPWCHG
```

NOREFRESH Keyword—Prevent Logons Copying Profile

Valid on z/OS.

Use the NOREFRESH keyword to prevent new logons allocating a new copy of the profile while updates are being made to the profile.

If NOREFRESH is set, when a profile is read during logon of a new user signing into a MUAS with a shared profile table, if there is an existing occurrence of the profile, CA Top Secret does not bring in the newer profile. This can be overridden by the PROFINTERVAL control option.

This keyword has the following format:

```
TSS [ADDTO|REMOVE] (profile) NOREFRESH
```

This keyword is used with:

- The commands ADDTO, and REMOVE
- The Profile ACID type
- ACID(CREATE) authority

NORESCHK Keyword—Bypass Security Checking

Valid on z/OS, z/VSE, and z/VM.

Use the NORESCHK to allow an ACID to bypass security checking for all owned resources except data sets and volumes. Auditing will occur.

This keyword has the following format:

```
TSS ADDTO(acid) NORESCHK
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC9(BYPASS) authority

Examples: NORESCHK keyword

This example allows BYUSER to bypass security checks for resource access (except data sets and volumes).

```
TSS ADDTO(BYUSER) NORESCHK
```

This example remove the NORESCHK attribute:

```
TSS REMOVE(BYUSER) NORESCHK
```

NOSUBCHK Keyword—Bypass Alternate Security Checking

Valid on z/OS, z/VSE, and z/VM.

Use the NOSUBCHK keyword to allow an ACID to bypass alternate ACID usage as well as all job submission security checking. Associated ACIDs may submit all jobs regardless of the (derived) ACID on the job card being submitted. Auditing will occur.

NOSUBCHK grants the ACID authorization to submit any batch job on behalf of another ACID without requiring a password. This includes the MSCA, emergency ACIDs, and system critical ACIDs. This attribute should be given out on a limited basis. If a schedular does its own cross submit authorization checking during the batch submit, it is safe to give the NOSUBCHK attribute to the schedular started task ACID. Otherwise, there is no way to audit the schedular's ACID.

This keyword has the following format:

```
TSS ADDTO(acid) NOSUBCHK
```

Note: For z/OS, the USER= keyword must still be inserted onto the job card prior to z/OS submission.

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC9(BYPASS) authority

Examples: NOSUBCHK keyword

This example allows a Production Controller to submit a job on behalf of any accessor:

```
TSS ADDTO(PRDCTL1) NOSUBCHK
```

This example removes the NOSUBCHK attribute:

```
TSS REMOVE(PRDCTL1) NOSUBCHK
```

NOSUSPEND Keyword—Bypass Violation Suspensions

Valid on z/OS, z/VSE, and z/VM.

Use the NOSUSPEND keyword to allow an ACID to bypass suspension due to violations (VTHRESH and PTHRESH). All actions coded in the VTHRESH and PTHRESH control options are suppressed for ACIDSs that have the NOSUSPEND attribute.

This keyword has the following format:

```
TSS ADDTO(acid) NOSUSPEND
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE (CREATE and REMOVE can only be issued for USER type ACIDs.)
- The ACID types User, DCA, VCA, SCA, LSCA, and MSCA
- MISC9(BYPASS) authority

Examples: NOSUSPEND keyword

The ACID assigned as the CICS default user is an example of an ACID shared by many users that should never suspend as a result of violations. This example allows CICS DFLTUSER=TOPSMAN to bypass being suspended due to violations:

```
TSS ADDTO(TOPSMAN) NOSUSPEND
```

This example removes the NOSUSPEND attribute:

```
TSS REMOVE(TOPSMAN) NOSUSPEND
```

NOVMDCHK Keyword—Bypass Minidisk Link Checking

Valid on z/VM.

Use the NOVMDCHK keyword to allow an ACID to bypass all checking for minidisk links. All links are audited.

NOVMDCHK is only applied to special products such as DASD space managers, which may link to many minidisks.

This keyword has the following format:

```
TSS ADDTO(acid) NOVMDCHK
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The NOVMDCHK The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC9(BYPASS) authority

Examples: NOVMDCHK keyword

This example allows the ACID DASDMAN to bypass all link verification (including link passwords) for any link:

```
TSS ADDTO(DASDMAN) NOVMDCHK
```

This example removes the NOVMDCHK attribute:

```
TSS REMOVE(DASDMAN) NOVMDCHK
```

NOVOLCHK Keyword—Bypass Volume Level Checking

Valid on z/OS, z/VSE, and z/VM.

Use the NOVOLCHK keyword to allow an ACID to bypass volume level security checking. Auditing will occur.

Note: NOVOLCHK does not necessarily give access to data sets on the volume. Administrators must enter the NODSNCHK keyword along with NOVOLCHK to allow total access to an entire volume when individual data sets are also being accessed.

This keyword has the following format:

```
TSS ADDTO(acid) NOVOLCHK
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC9(BYPASS) authority

Examples: NOVOLCHK keyword

This example assigns the NOVOLCHK attribute to an ACID within his scope:

```
TSS ADDTO(USER01) NOVOLCHK
```

This example allows use of volumes that are not globally allowed for data set creation:

```
TSS ADDTO(USER01) NOVOLCHK
                  NODSNCHK
```

This example removes the NOVOLCHK attribute:

```
TSS REMOVE(USER01) NOVOLCHK
```

OECPUTM Keyword—Maximum CPUTIME for a Dubbed Process

Valid on z/OS.

Use the OECPUTM keyword to specify the maximum time in seconds a process is allowed to use. This value overrides the MAXCPUTIME parameter in the BPXPRMxx member of PARMLIB for this user.

OECPUTM is equivalent to CPUTIMEMAX in RACF

This keyword has the following format:

```
TSS ADD(acidname) OECPUTM(nnnnnnnn)
```

nnnnnnnnn

The maximum time in seconds a process is allowed to use.

Range: 7 to 2,147,483,647

Default: USS takes the system defaults set in BPXPRMxx

This keyword is used with the ADDTO, REPLACE, REMOVE, and WHOHAS commands.

Examples: OECPUTM keyword

This example assigns an acid a value of 10 seconds for OECPUTM:

```
TSS ADD(TESTID) OECPUTM(10)
```

This example removes OECPUTM from the acid:

```
TSS REMOVE(TESTID) OECPUTM
```

OEFILEP Keyword—Maximum Files per Process

Valid on z/OS.

Use the OEFILEP keyword to specify the maximum number of files that a single process can have active or open concurrently. This field overrides the MAXFILEPROC parameter in the BPXPRMxx.

OEFILEP is equivalent to FILEPROCMAX in RACF.

This keyword has the following format:

```
TSS ADD(acidname) OEFILEP(nnnnn)
```

nnnnn

The maximum number of files that a single process can have active or open concurrently.

Range: 3 to 65,535

Default: USS takes the system defaults set in BPXPRMxx

This keyword is used with the ADDTO, REPLACE, REMOVE, and WHOHAS commands.

Examples: OEFILEP keyword

This example assigns an acid a value of 20 for OEFILEP:

```
TSS ADD(TESTID) OEFILEP(20)
```

This example removes OEFILEP from the acid:

```
TSS REMOVE(TESTID) OEFILEP
```

OIDCARD Keyword—Prompt for Identity Card

Valid on z/OS and z/VM.

Use the OIDCARD keyword to prompt ACIDs to insert their identification cards into a batch reader whenever they sign on to TSO and z/VM. In CICS, CA Top Secret ACIDs are not prompted.

This keyword has the following format:

```
TSS ADDTO(acid) OIDCARD
```

This command result in a TSS0298A message, “INSERT OPERATOR ID CARD.” The administrator must be at the badge reader when he enters the TSS command to add the OID to an ACID's Security Record. Once the user's information has been added to the Security Record, the user is required to insert his OIDCARD at each signon.

Prompting for the operator identification card value is only possible through TSO and z/VM. In CICS, the ACID is not prompted but must have the OIDCARD inserted at signon.

This keyword is used with:

- The commands CREATE, REPLACE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: OIDCARD keyword

This example forces CUST800 to present his OIDCARD when he signs on:

```
TSS ADDTO(CUST800) OIDCARD
```

This example removes the OIDCARD attribute:

```
TSS REMOVE(CUST800) OIDCARD
```

OMVSPGM Keyword—Add or Remove a Program

Valid on z/OS and z/VM.

Use the OMVSPGM keyword to add or remove a program from a specified ACID.

This keyword has the following format:

```
TSS ADDTO(acid) OMVSPGM(programname)
```

Capacity of list

One 1024 byte Open OS/390 program name.

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

This keyword is used with the ADDTO and WHOHAS commands.

Examples: OMVSPGM keyword

This example indicates that program AABESS0099 is now connected to USER02:

```
TSS ADDTO(USER02) OMVSPGM(AABESS0099)
```

This example removes the connection between a program and an ACID:

```
TSS REMOVE(USER02) OMVSPGM(AABESS0099)
```

ONAME1 Keyword—Data Classification Owner Data

Valid on z/OS.

This optional field is used to allow up to 255 bytes of comments regarding OWNER1's ownership of the record. This keyword is part of the DATACLAS tracking record that helps determine which data (files, data sets, resources) pertains to which regulation. This record resides in the VSAM file.

ONAME1 can contain letters, numbers, or special characters. If the ONAME1 data contains blanks, place quotation marks around the data.

Keyword length

One to 255 characters.

Capacity of list

One data classification ONAME1 field per command.

This keyword is used with:

- The commands ADDTO and REMOVE
- The DATACLAS record only
- MISC5(DCLADMIN) authority
- MISC5(DCLLIST) authority to LIST data classes in the DATACLAS record

Example: ONAME1 Keyword

This example adds a HIPAA DATACLAS record:

```
TSS ADDTO(DATACLAS) DCLASS(HIPAA)
                    RCLASS(DATASET)
                    RESNAME(HIPAA.RECORDS)
                    OWNER1(HIPOWNER)
                    ONAME1('HIPOWNER owns all HIPAA datasets')
```

ONAME2 Keyword—Data Classification Owner Data

Valid on z/OS.

This optional field is used to allow up to 255 bytes of comments regarding OWNER2's ownership of the record. This keyword is part the DATACLAS tracking record that helps determine which data (files, data sets, resources) pertains to which regulation. This record resides in the VSAM file.

ONAME2 can contain letters, numbers, or special characters. If the ONAME2 data contains blanks, place quotation marks around the data.

Keyword length

One to 255 characters.

Capacity of list

One data classification ONAME2 field per command.

This keyword is used with:

- The commands ADDTO and REMOVE
- The DATACLAS record only
- MISC5(DCLADMIN) authority
- MISC5(DCLLIST) authority to LIST data classes in the DATACLAS record

Example: ONAME2 Keyword

This example adds resource class DATASET to the HIPAA DATACLAS record:

```
TSS ADDTO(DATACLAS) DCLASS(HIPAA)
                    RCLASS(DATASET)
                    RESNAME(HIPAA.RECORDS)
                    OWNER1(HIPDCA)
                    OWNER2(HIPOWNER)
                    ONAME2('HIPOWNER owns all HIPAA datasets')
```

OPCLASS Keyword—Maintain CICS Operator Classes

Valid on z/OS and z/VSE.

Use the OPCLASS keyword to assign or remove a set of CICS operator classes. These values replace those normally found in an ACID's SNT (Signon Table) entry. The OPCLASS values are placed into an ACID's TCT (Terminal Control Table) entry at signon.

This keyword has the following format:

```
TSS ADDTO(acid) OPCLASS(nn,nn..)
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: OPCLASS keyword

This example adds OPCLASS values to user CLRK55:

```
TSS ADDTO(CLRK55) OPCLASS(1,4,7)
```

This example removes a user's OPCLASS values:

```
TSS REMOVE(CLRK55) OPCLASS(1,4,7)
```

OPIDENT Keyword—Maintain CICS Operator Identification

Valid on z/OS and z/VSE.

Use the OPIDENT keyword to assign or remove a CICS operator identification value equal to the ACID's OPIDENT entry in the CICS SNT (Signon Table). The OPIDENT value is placed into the ACID's TCT at signon.

This keyword has the following format:

```
TSS ADDTO(acid) OPIDENT(xxx)
```

Prefix length

One to three characters

This keyword is used with:

- The commands CREATE, REPLACE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: OPIDENT keyword

This example assigns an OPIDENT to user BLAS19:

```
TSS ADDTO(BLAS19) OPIDENT(XYZ)
```

This example removes the user's OPIDENT:

```
TSS REMOVE(BLAS19) OPIDENT
```

Note: Do not embed blank spaces to substitute characters. The following is not a valid entry.

```
TSS ADDTO(BLAS19) OPIDENT(X Y)
```

OPPRTY Keyword—Maintain CICS Operator Priority

Valid on z/OS and z/VSE.

Use the OPPRTY keyword to assign or remove a CICS operator priority from the associated ACID. The OPPRTY value is placed into the ACID's TCT (Terminal Control Table) at signon.

This keyword has the following format:

```
TSS ADDTO(acid) OPPRTY(0...255)
```

Prefix length

0 to 255 (where 0 is no special priority).

This keyword is used with:

- The commands CREATE, REPLACE, ADDTO, REMOVE, and WHOHAS
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: OPPRTY keyword

This example assigns an OPPRTY of 10 to USER01:

```
TSS ADDTO(USER01) OPPRTY(10)
```

This example removes the OPPRTY assignment:

```
TSS REMOVE(USER01) OPPRTY(10)
```

OPTIME keyword—Maintain CICS Timeout

Valid on z/OS

Use the OPTIME keyword to assign or remove a CICS timeout value, which is equivalent to CICS TIMEOUT functionality. During user signon, CA Top Secret places the OPTIME value in the TCTTE of the user.

This keyword has the following format:

```
TSS ADDTO(acid) OPTIME(hhmm)
```

hh

Defines hours until timeout.

mm

Defines minutes until timeout.

This keyword is used with:

- The commands CREATE, REPLACE, ADDTO, and REMOVE
- The acid type User, Profile, DCA, VCA, ZCA, LSCA, SCA, MSCA

Examples: OPTIME keyword

This example assigns a timeout time of 30 minutes to user andy01:

```
TSS ADDTO(andy01) OPTIME(30)
```

This example removes the user's timeout time:

```
TSS REMOVE(andy01) OPTIME
```

OWNER1 Keyword—Data Classification Owner Definition

Valid on z/OS.

Use the OWNER1 keyword to define a Data Classification resource owner ACID. This keyword is part the DATACLAS tracking record that helps determine which data (files, data sets, resources) pertains to which regulation. This record resides in the VSAM file.

Keyword length

One to eight characters.

Capacity of list

One data classification resource class per command.

This keyword is used with:

- The commands LIST, ADDTO, and REMOVE
- The DATACLAS record only
- MISC5(DCLADMIN) authority
- MISC5(DCLLIST) authority to LIST data classes in the DATACLAS record

Examples: OWNER1 Keyword

This example adds resource class DATASET to the HIPAA DATACLAS record:

```
TSS ADDTO(DATACLAS) DCLASS(HIPAA)
                    RCLASS(DATASET)
                    RESNAME(HIPAA.RECORDS)
                    OWNER1(HIPOWNER)
```

This example removes data class HIPAA records with resource DATASET resource class from the DATACLAS record:

```
TSS REMOVE(DATACLAS) DCLASS(HIPAA)
                     RCLASS(DATASET)
                     RESNAME(HIPAS.RECORDS)
                     OWNER1(HIPOWNER)
```

Note: To remove a record, the DCLASS, RCLASS, RESNAME, OWNER1, and OWNER2 fields must match exactly to list output.

This example lists data concerning data class HIPAA and owned by HIPOWNER:

```
TSS LIST(DATACLAS) DCLASS(HIPAA)
                   OWNER1(HIPOWNER)
```

OWNER2 Keyword

Valid on z/OS.

Use the OWNER2 keyword to define a Data Classification resource owner ACID. This keyword is part the DATACLAS tracking record that helps determine which data (files, data sets, resources) pertains to which regulation. This record resides in the VSAM file.

You cannot specify OWNER2 without OWNER1.

Keyword length

One to eight characters.

Capacity of list

One data classification resource class per command.

This keyword is used with:

- The commands LIST, ADDTO, and REMOVE
- The DATACLAS record only
- MISC5(DCLADMIN) authority
- MISC5(DCLLIST) authority to LIST data classes in the DATACLAS record

Examples: OWNER2 keyword

This example adds resource class DATASET to the HIPAA DATACLAS record:

```
TSS ADDTO(DATACLAS) DCLASS(HIPAA)
                    RCLASS(DATASET)
                    RESNAME(HIPAA.RECORDS)
                    OWNER1(HIPOWNER)
                    OWNER2(HIPDCA)
```

This example removes data class HIPAA records with resource DATASET resource class from the DATACLAS record:

```
TSS REMOVE(DATACLAS) DCLASS(HIPAA)
                     RCLASS(DATASET)
                     RESNAME(HIPAS.RECORDS)
                     OWNER1(HIPOWNER)
                     OWNER2(HIPDCA)
```

Note: To remove a record the CLASS, RCLASS, RESNAME, OWNER1, OWNER2 fields must match exactly to list output..

This example lists data concerning data class HIPAA and owned by HIPOWNER:

```
TSS LIST(DATACLAS) DCLASS(HIPAA)
                   OWNER1(HIPOWNER)
```

PASSWORD Keyword (for Users)—Change a Password

Valid on z/OS, z/VSE, and z/VM.

Non-administrative ACIDs can use the PASSWORD keyword to change their own passwords (as long as the NOPWCHG attribute is not assigned and the new password conforms to the NEWPW control option restrictions).

If the ACID has multiple passwords assigned by facility, the MULTIPW and FACILITY options must be included to identify the specific facility password to change. If these options are not included on the password change request, the change applies to the default facility *ALL*.

Passwords entered through the Administrative Panels are "invisible." Passwords entered through TSS commands are entered as text.

Some teleprocessing monitors automatically convert mixed-case passwords to uppercase before processing by CA Top Secret. These monitors cannot match a mixed-case password.

When used by a non-administrative ACID, this keyword has the following format:

```
TSS REPLACE(acid) PASSWORD([old_password/]new_password)
                  [FACILITY(facility) MULTIPW]
```

old_password

(Optional) Specifies the existing password string that you want to change. If supplied, the old password must match the user's password and must be separated from the new password by a slash ("/").

new_password

Specifies the new password string for the ACID. The new password must meet NEWPW control option requirements.

FACILITY

(Optional) Supplies a facility with the MULTIPW keyword, which means that the password applies only to the specific facility. Specifying a MULTIPW facility password allows the administrator to account for variations in mixed-case and extended-length passwords.

Separate password history is provided for each MULTIPW facility password.

MULTIPW

(Optional) Confirms that the password applies only to the specific facility. When a non-administrative ACID assigns a new password in a facility where an administrator assigned a MULTIPW facility password, the syntax must include the MULTIPW and FACILITY options to target the correct facility for the change.

Non-administrative ACID types can specify the PASSWORD keyword only on the ADD or REPLACE command. These ACID types can change the password for users within their scope if they have UPDATE access to entity TSSCMD.USER.*cmd*.PASSWORD in the CASECAUT resource class.

Note: For details about using CASECAUT to provide restricted administrative authorities, see the *CA Top Secret User Guide*.

Example: Replace the Password for the CICSPROD Facility

This example shows a non-administrator, multi-password ACID (USER56) replacing the password for the facility CICSPROD:

```
TSS REPLACE(USER56) PASSWORD(WORK)
                    MULTIPW
                    FACILITY(CICSPROD)
```

PASSWORD Keyword (for Administrators)—Assign a Password

Valid on z/OS, z/VSE, and z/VM.

An administrative ACID can use the PASSWORD keyword to assign a password, along with values that control its use, to a previously defined ACID.

Note: An administrative ACID can assign passwords that do not conform to NEWPW control option restrictions; however, passwords must not exceed the MAX length set for the system. For z/OS, the administrator may set control option PWADMIN(YES) to enforce NEWPW control option restrictions and expiration interval restrictions on administrators.

Passwords entered through the Administrative Panels are "invisible." Passwords entered through TSS commands are entered as text.

Some teleprocessing monitors automatically convert mixed case passwords into uppercase before processing by CA Top Secret. These monitors cannot match a mixed case password.

A password that is specified by the ADDTO command function replaces the ACID's previous password.

When used by an administrative ACID, this keyword has the following format:

```
TSS ADDTO(acid) PASSWORD(password[,[expiration_interval][,EXP]])
                [FACILITY(facility) MULTIPW]

TSS ADDTO(acid) PASSWORD(NOPW)
                [FACILITY(facility) MULTIPW]
```

password

Specifies one of the following values:

- a one- to eight-character password string, as limited by control options NEWPW, PWADMIN, and PASSCHAR.
- * (specifies that the password string should remain unchanged by this command). This value is the default.

expiration_interval

(Optional) Specifies an interval (days) for the current password before it expires (0 indicates a non-expiring password).

Range: 0 to 255

If not included in the PASSWORD keyword specification for ADD or CREATE, the expiration interval defaults to the PWEXP control option value. For example, the expiration interval is not specified in the following commands:

```
TSS ADDTO(acid) PASSWORD(pass1)
TSS ADDTO(acid) PASSWORD(pass1,,EXP)
```

If not specified in REPLACE (as shown in the following example), the current expiration interval is retained:

```
TSS REP(acid) PASSWORD(pass1)
```

EXP

(Optional) Indicates that the password string is issued as an expired password and must be immediately changed.

NOPW

Specifies that no password is required for this ACID.

Note: Setting this value requires UPDATE access to entity TSSCMD.USER.*cmd*.NOPW in the CASECAUT recourse class. For complete information about CASECAUT, see the *CA Top Secret User Guide*.

FACILITY

Specifies that the password applies only to the specific facility (FACILITY must be supplied with the MULTIPW keyword). Specifying a MULTIPW facility password allows the administrator to account for variations in mixed-case and extended length passwords.

Separate password history is provided for each MULTIPW facility password.

MULTIPW

Confirms that the password applies only to the specific facility. This keyword is required when a new password is added by an administrative ACID for a designated facility.

When a non-administrative ACID assigns a new password in a facility where an administrator assigned a MULTIPW facility password, they must include the MULTIPW and FACILITY options to target the correct facility for the change.

This keyword is used with:

- The commands CREATE, REPLACE, and ADDTO
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

  Administrators who do not have ACID(MAINTAIN) authority can add a password to users within their scope if given UPDATE access to entity TSSCMD.USER.ADDTO.PASSWORD in the CASECAUT resource class.

  Administrators who do not have ACID(MAINTAIN) authority can replace a password for users within their scope if given UPDATE access to entity TSSCMD.USER.REPLACE.PASSWORD in the CASECAUT resource class.

Other requirements are as follows:

- When setting a NOPW password value (using CREATE, ADDTO or REPLACE), administrators must have UPDATE access to entity TSSCMD.USER.*cmd*.NOPW in the CASECAUT resource class, where *cmd* is the command being issued.
- To set a new password value for the MSCA (using ADDTO or REPLACE), an SCA must must have UPDATE access to entity TSSCMD.USER.*cmd*.MSCAPW in the CASECAUT resource class, where *cmd* is the command being issued.

  Note: Always replace *cmd* with the full command name.

Example: Replace a Password and Require Immediate Password Change

This example replaces USER56's previous password with WORK and prompts the user to change the password immediately during that user's first signon. USER56 is forced to change passwords every nine days.

```
TSS ADDTO(USER56) PASSWORD(WORK,9,EXP)
```

Example: Assign a Password

In this example, USER56 signs on with this password until it expires in 30 days (or the default set by the PWEXP control option) or until USER56 changes it. This example assumes that the NEWPW control option is not set to NU to prevent USER56 from changing his or her own password.

```
TSS ADDTO(USER56) PASSWORD(WORK)
```

Example: Replace a Password

This example shows a non-administrator (USER56) with the default NEWPW option set of MIN=4,MAX=8,MINDAY=1,WARN=3:

```
TSS REPLACE(USER56) PASSWORD(ALLEN11)
```

If the NEWPW option NR=1 is set, the above password change fails.

Example: Replace an Expiration Interval

This example replaces USER01's expiration interval without changing or knowing USER01's password:

```
TSS ADDTO(USER01) PASSWORD(*,10)
```

PCICC Keyword—Generate Keys with PCI Cryptographic Coprocessor

Valid on z/OS.

Use the PCICC keyword to specify that the key pair is generated using the PCI Cryptographic Coprocessor, and that the private key is stored in ICSF PKDS. When PCICC is not specified, the key pair is generated using software. PCICC cannot be used with the DSN, DSA, or ICSF parameters. If a PCI cryptographic coprocessor is not present or operational, or if ICSF is not active or configured for PKA operations, an error message is displayed and processing terminates.

If ICSF, PCICC, or LABLPKDS is not specified, software generates the key pair and the private key is stored in the security file as a non-ICSF key.

This keyword has the following format:

```
TSS REKEY(acid) DIGICERT(8-byte name)
                DCDSN(dsname)
                PCICC
```

This keyword is used with:

- The commands ADDTO, GENCERT, REKEY and RENEW
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority for users
- MISC4(CERTUSER) authority for user ACIDs
- MISC4(CERTSITE) authority for CERTSITE ACIDs
- MISC4(CERTAUTH) authority for CERTAUTH ACIDs

Examples: PCICC keyword

This example indicates that the digital certificate is stored in an PCICC facility:

```
TSS ADDTO(USERA) DIGICERT(USERACER)
                 DCDSN(USERA.CERT01)
                 PCICC
```

This example removes the entry in the PCICC storage facility by removing the DIGICERT from the acid:

```
TSS REMOVE(USERA) DIGICERT(USERACER)
```

PHRASE Keyword—Assign Password Phrase

Valid on z/OS 1.8 and above.

Use the PHRASE keyword to assign a password phrase as an alternative to PASSWORD.

If PHRASE is used to signon using RACROUTE VERIFY VERIFYX or USS initACEE, the PASSWORD and NEWPASSWORD keywords are ignored.

If PASSWORD is used to signon without PHRASE then NEWPHRASE is ignored.

The text in a password phrase can be composed of:

- Upper and lower case alphabetical characters
- Numeric characters
- Special characters included in the PPSCHAR control option

Special characters can be used regardless of whether:

- The security file is copied with NEWPWBLOCK
- NEWPW(MC) is set

There is no analog for PHRASE to MULTIPW for facility specific phrases.

This keyword has the following format:

```
TSS ADDTO(acid) PHRASE(phrase[,nnn][,EXPIRED])
```

phrase

Specify a quoted string delimited by apostrophes. For restrictions on user supplied password phrases, see the NEWPHRASE control option.

Specify an asterisk (*) to change the expiration interval or expired indicator keyword (EXP) without changing the existing password phrase.

Range: 9 to 100 characters

nnn

(Optional) Specifies the password phrase expiration interval. When REPLACE is used to change the password phrase and no interval is specified, the user's existing password phrase interval is retained.

Range: 0 to 255

Default: Set by the PPEXP control option (30 days)

EXPIRED

(Optional) Forces an expiration date of January 1, 1980, regardless of the date the command is executed. This forces the PHRASE into expired status and the user to enter a new password phrase in the next signon which occurs with a password phrase but without a password.

This keyword is used with:

- The commands CREATE, REPLACE, and ADDTO
- ACID(MAINTAIN) authority
- Administrators who do not have ACID(MAINTAIN) authority can add a password PHRASE to users within their scope if given UPDATE access to entity TSSCMD.USER.ADDTO.PHRASE in the CASECAUT resource class
- Administrators who do not have ACID(MAINTAIN) authority can replace a password PHRASE for users within their scope if given UPDATE access to entity TSSCMD.USER.REPLACE.PHRASE in the CASECAUT resource class

PHYSKEY Keyword—Support External Authentication Devices

Valid on z/OS.

Use the PHYSKEY (physical security key) keyword to support external authentication devices.

This keyword has the following format:

```
TSS ADDTO(acid) PHYSKEY(['])physkey data['])
```

PHYSKEY data:

- May contain from one to 255 characters
- Must be delimited by single quotes if it contains blanks, lower case, or special characters
- Is automatically uppercased unless OPTIONS(73 or 16) were set during TSS startup

This keyword is used with:

- The commands CREATE, REPLACE, ADDTO, and REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: PHYSKEY keyword

This example specifies that CUST800 has PHYSKEY 4739:

```
TSS ADDTO(CUST800) PHYSKEY(4739)
```

This example removes CUST800's PHYSKEY:

```
TSS REMOVE(CUST800) PHYSKEY(4739)
```

PKCSPASS Keyword—Specify Certificate Password

Valid on z/OS.

Use the PKCSPASS keyword to specify the password associated with a PKCS#12-formatted digital certificate. A password is required if the data set contains a PKCS#12-format certificate that is password protected.

Notes:

- The password can be up to 255 characters, is case sensitive, and can contain blanks.
- The CA Top Secret Administrator is responsible of keeping track of the PKCSPASS password they assign to a digital certificate. There is no way to list this password online.
- If the certificate's private key resides in an ICSF storage facility and the format of PKCS12DER or PKCS12B64 is specified in the TSS EXPORT command, the command is rejected and a TSS0533E message is issued.
- If ICSF is specified and the IBM ICSF feature is enabled, the private key is stored in the ICSF data facility.
- ICSF is the interface to the cryptographic hardware on z/OS. You must have cryptographic hardware installed and enabled on your system.

This keyword has the following format:

```
TSS CHKCERT DCDSN(input-data-set-name)
            PKCSPASS(pkcs#12-password)
```

This keyword can be used with:

- The commands ADDTO, CHKCERT, EXPORT, and GENCERT
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- MISC4(CERTCHEK) and MISC4(CERTSITE) authority for CERTSITE ACIDs
- MISC4(CERTAUTH) authority for CERTAUTH ACIDs

Example: PKCSPASS

This example sets the password to 'J Doe12345 Pwnew".

```
TSS CHKCERT DCDSN(user1.cert.data)
            PKCSPASS('J Doe12345 Pwnew')
```

POSIT Keyword—Class POSIT Number

Valid on z/OS.

Use the POSIT keyword to specify the POSIT number associated with the class. The POSIT identifies a unique set of options associated with a resource class. There are 1024 POSIT numbers that can identify 1024 sets of option flags. Installations can specify POSIT numbers 19-56 and 128-527. Numbers 0-18, 57-127, and 528-1023 are reserved for IBM use.

This operand may only be specified for user resource classes.

This keyword has the following format:

```
TSS ADD(RDT) RESCLASS(resource class)
             RESCODE(hex code)
             POSIT(nnnn)
```

nnnn

Is a decimal value between 19—56, and 128—527.

The keyword is used with:

- The commands ADDTO and REPLACE
- The RDT record only
- MISC1(RDT) authority

Example: POSIT keyword

This example defines an RDT POSIT value of 23 for the user resource class:

```
TSS REPL(RDT) RESCLASS(HRES) POSIT(23)
```

PREFIX Keyword—Record Matching by Partial Key

Valid on z/OS.

Use the PREFIX keyword to specify that all record matching is performed using the data entered as a partial key.

This keyword has the following format:

```
TSS LIST(STC) PROCNAME(ASSEM) PREFIX
TSS LIST(SDT)
TSS LIST(RDT)
TSS LIST(FDT)
```

Examples: PREFIX keyword

This example lists the FDT table using the PREFIX option:

```
TSS LIST(FDT) FDTNAME(xxxxxxxx) PREFIX
```

This example lists the RDT table using the prefix option:

```
TSS LIST(RDT) RESCLASS(xxxxxxxx) PREFIX
```

This example lists the STC table using the prefix option:

```
TSS LIST(STC) PROCNAME(xxxxxxxx)PREFIX
```

```
TSS LIST(STC) ACID(xxxxxxxx)PREFIX
```

```
TSS LIST(STC) PROCNAME(xxxxxxxx)ACID(xxxxxxxx)PREFIX
```

This example lists the SDT table using the prefix option:

```
TSS LIST(SDT) TIMEREC(xxxxx) PREFIX
```

PRIVPGM Keyword—Specify Programs in Control

Valid on z/OS.

Use the PRIVPGM keyword to specify the full names of the programs that must be in control when a data set is accessed. A privileged program specification restricts the use of a resource to a control path.

This keyword has the following format:

```
TSS PERMIT(acid) DSN(dataset)|FCT(fct_dataset) PRIVPGM(program)
```

acid

Specifies the ACID to which you are permitting access to a data set.

dataset|fct_dataset

Specifies the data set name or data set prefix to which the ACID is gaining access. An FCT is a data set that is accessed through CICS.

Prefix length: One to 44 characters

program

Specifies the programs that must be in control when a data set is accessed. Any specified prefixes must include (G) at the end of the prefix.

Prefix length: One to eight characters

Capacity of list: One to eight program names

(DSNAMEs only) PRIVPGM may be combined with LIBRARY to specify the library in which the privileged program must reside. If LIBRARY is not specified, the privileged program must come from the link pack area (LPA) or from a library in the link list.

The privileged program specified in the PERMIT function must be executed through one of the following methods:

- JCL keyword EXEC PGM=*program*
- A program executed through LINK, LOAD, ATTACH, or XCTL

Technically, the program must be associated with the top-most or bottom-most Program Request Block (PRB) in z/OS.

This keyword is used with:

- The PERMIT command
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ALL
- Resource(XAUTH) authority (required to specify PRIVPGM for resources)

Example: Permit Access to SMF Data Sets through a Link-Listed Program

This example permits access to SMF data sets SYS1.MANX and SYS1.MANY specifically through a link-listed program:

```
TSS PERMIT(SYS92) ACCESS(CONTROL)
                  DSNAME(SYS1.MAN+)
                  PRIVPGM(SMFMNGR)
```

Example: Permit Access to a Data Set through Program PRSP60

This example permits access to a personnel data set only if program PRSP60 is used and is loaded from a specific library:

```
TSS PERMIT(PAY100K) DSNAME(PERS.*.MASTER)
                    PRIVPGM(PRSP60)
                    LIBRARY(PERS.LOADLIB)
```

Example: Permit Access to a Data Set through Any Program Beginning with PRSP

This example permits access to the same personnel data set but allows any program beginning with the characters PRSP:

```
TSS PERMIT(PAY100K) DSNAME(PERS.*.MASTER)
                    PRIVPGM(PRSP(G))
```

Example: Permits Access to a CICS file through CICS Program PCC505

This example permits access to a CICS file only if the CICS program in control is PCC505:

```
TSS PERMIT(CLK505P) FCT(PARTS)
                    PRIVPGM(PCC505)
```

PROCNAME Keyword—Define STC Procedure

Valid on z/OS.

Use the PROCNAME keyword to define a started task to CA Top Secret by associating the STC procedure name with an ACID or action.

An STC procedure can be generically defined to the STC table. Use an asterisk at the end of an STC procedure entry to signify that the stcname is considered a prefix at run-time. If multiple STC entries match an STC, CA Top Secret selects the best match based on the following selection criteria:

- Exact match of non-generic entry
- Longest matching generic entry
- If there is no matching generic entry, the default STC ACID is used
- If there is no DEFAULT stcname, the STC FACILITY DEFACID is assigned
- If no ACID can be determined, the MODE of the STC facility determines the disposition of the started task

For information, see the *User Guide* and *Implementation: Batch, STC, and APPC Guide*.

The STCACT attribute can force the operator to identify the ACID and PASSWORD that entered the START command for the associated PROCNAME. The signon of this operator accountability ACID generates a logged event. STC AccounTability (STCACT) when called for (message TSS7152A) provides:

- A separate identification prior to prompting the operator for an optional execution ACID (PROMPT, resulting in TSS7151A)
- An execution PASSWORD (when the ACID of the STC definition contains a password other than NOPW, resulting in TSS7150A)

For information, see the STCACT keyword.

This keyword has the following format:

```
TSS ADDTO(STC) PROCNAME(stcname|DEFAULT)
               ACID(acid|action)
               [STCACT]
```

stcname

The STC procedure name.

Range: 1 to 8 characters

DEFAULT

Provides undefined started tasks with a default ACID or default action. A started task is undefined if the procname in the operator console START command does not match any of the STC PROCNAME procedures or prefixes on file.

acid

Pre-existing ACID that CA Top Secret assigns the started task with designated PROCNAME. If the ACID is defined with a significant password (other than NOPW), the system console prompts for the PASSWORD associated with "acid." If the ACID is deleted, if the stcname procedure is started and assigned to this now invalid ACID. An error is issued by the delete but the command is allowed to continue.

Range: 1 to 8 characters

action

If no ACID was created for the STC, choose one of the following actions:

- BYPASS—Security checking for the stcname is bypassed. This is not recommended for complex tasks like CICS, where BYPASS security can cause security failure and unpredictable outcomes during initialization and transaction execution.
- FAIL—The stcname initiation fails.
- PROMPT—The console operator is prompted for an ACID and password assigned for this instance of the started task of this stcname.

STCACT

When this keyword is present on an ADD command, the console operator is prompted for an accountability ACID and password, responsible for entering the START operator command. This ACID is separate from the ACID assigned to execute the started task provided by the ACID operand: it provides a logged event for accountability of the START command.

This keyword is used with:

- The commands ADDTO, REMOVE, and LIST
- The STC special record only
- MISC9(STC) authority

Examples: PROCNAME keyword

This example fails all undefined STCs:

```
TSS ADDTO(STC) PROCNAME(DEFAULT)
               ACID(FAIL)
```

This example allow undefined STCs to BYPASS security:

```
TSS ADDTO(STC) PROCNAME(DEFAULT)
               ACID(BYPASS)
```

This example associates undefined STCs with a selected pre-existing ACID U123456:

```
TSS ADDTO(STC) PROCNAME(DEFAULT)
               ACID(U123456)
```

This example forces an operator to assign the ACID and password assigned to each separate execution of STC WHOAREU:

```
TSS ADDTO(STC) PROCNAME(WHOAREU)
               ACID(PROMPT)
```

This example forces an operator to indicate who started task WHOSTRT and assigns an execution ACID of MESTRT:

```
TSS ADDTO(STC) PROCNAME(WHOSTRT)
               ACID(MESTRT)
               STCACT
```

When executed, the following prompts appear:

```
S STCATC1
04 TSS7152A Specify Your AccessorID/Password to Allow Use of STC=STCATC1
R 4,OPER1A/SUPER
IEE600I REPLY TO 04 IS;SUPPRESSED
05 TSS7150A SPECIFY PASSWORD FOR STC=STCATC1 ACID=STCATC1
R 5,STC
IEE600I REPLY TO 05 IS;SUPPRESSED
$HASP100 STCATC1 ON STCINRDR
IEF695I START STCATC1 WITH JOBNAME STCATC1 IS ASSIGNED TO USER STCATC1
$HASP373 STCATC1 STARTED
IEF403I STCATC1 - STARTED - TIME=17.15.59
IEF404I STCATC1 - ENDED - TIME=17.16.00
```

Message TSS7150A appears due to the ACID operand, assigning ACID STCATC1 to PROCNAME STCATC1. For purposes of discussion, this ACID is listed below:

```
ACCESSORID = STCATC1 NAME = STCATC1
TYPE = USER SIZE = 256 BYTES
FACILITY = STC
DEPT ACID = SYSDEPT DEPARTMENT = QA SYSTEMS RESOUREC DEPT
```

```
CREATED = 04/20/06 LAST MOD = 04/20/06 12:15
LAST USED = 05/17/06 17:15 CPU(XE18) FAC(STC ) COUNT(00003)
PASSWORD = STC
```

The correct response to TSS7150A is the password of the assigned ACID.

The message TSS7152A appears in response to the STCACT keyword on the ADD command for PROCNAME STCATC1. The response provides a user and password indication of who entered the START command.

This example provides a common ACID CTSREGN for all procedure names that begin with the characters "CTS". CTS22 is set to FAIL because it is obsolete:

```
TSS ADD(STC) PROCNAME(CTS*)
             ACID(CTSREGN)

TSS ADD(STC) PROCNAME(CTS22)
             ACID(FAIL)
```

This example console command assigns the procname CTS31 to ACID CTSREGN:

```
START CTS31
```

This example console command fails:

```
START CTS22
```

These example STC definitions use invalid generic names:

```
TSS ADD(STC) PROCNAME(CTS*2)
             ACID(CICSYS2)

TSS ADD(STC) PROCNAME(*)
             ACID(TOOGENR)
```

This example removes definitions for STC PROCNAMES:

```
TSS REM(STC) PROCNAME(STCATC1)
```

This example associates a started task for a dump processing with an ACID (use the CREATE function to define ACID OP187):

```
TSS ADDTO(STC) PROCNAME(PRDMP)
               ACID(OP187)
```

This example associates undefined STCs with a default ACID (use the CREATE function to define ACID 12347):

```
TSS ADDTO(STC) PROCNAME(DEFAULT)
               ACID(12347)
```

PROCUSER Keyword—Maximum Number of Processes

Valid on z/OS.

Use the PROCUSER keyword to specify the maximum number of processes a user can have open at the same time.This field overrides the MAXPROCUSER parameter in the BPXPRMxx member of PARMLIB for this user.

PROCUSER is equivalent to PROCUSERMAX in RACF.

This keyword has the following format:

```
TSS ADD(acidname) PROCUSER(nnnnn)
```

***n*nnnn**

The maximum number of processes a user can have open at the same time.

Limits: 3 to 32,767

Default: USS takes the system defaults set in BPXPRMxx.

This keyword is used with the ADDTO, REPLACE, REMOVE and WHOHAS commands.

Examples: PROCUSER keyword

This example assigns an acid a value of 3 for PROCUSER:

```
TSS ADD(TESTID) PROCUSER(3)
```

This example removes PROCUSER from the acid:

```
TSS REMOVE(TESTID) PROCUSER
```

PROFILE Keyword—Maintain Profiles

Valid on z/OS and z/VM.

Use the PROFILE keyword to add or remove up to 254 profiles from a specified ACID.

Notes: The PROFILE and GROUP keywords:

- Cannot be used in the same TSS commands
- Are interchangeable and coding PROFILE() removes both GROUPS and PROFILE

This keyword has the following format:

```
TSS ADDTO(acid) PROFILE(profile acid,profile acid,...)
```

acid length

One to eight characters

Capacity of list

Five profile acids per TSS command

This keyword is used with:

- The commands CREATE ADDTO, and REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: PROFILE keyword

This example indicates that profile WRIT01 is now connected to USER02:

```
TSS ADDTO(USER02) PROFILE(WRIT01)
```

Administrators can designate the order of a new profile using the AFTER or BEFORE keywords with the profile name.

This example indicates that PROF03 should follow PROF01:

```
TSS ADDTO(acid) PROFILE(PROF03)
                AFTER(PROF01)
```

The FIRST keyword enables an administrator to add a profile to the beginning of a profile list. This example adds the profile FINPROF1 as the first one in USER01's profile list:

```
TSS ADDTO(USER01) PROFILE(FINPROF1)
                  FIRST
```

If you use the FOR or UNTIL keywords with PROFILE, you are able to see the expiration date(s):

```
TSS LIST(acid) DATA(EXP)
```

This example removes the connection between a profile and an ACID:

```
TSS REMOVE(USER02) PROFILE(WRIT01)
```

PROGRAM Keyword—Specify SIGVER record program definition

Valid on z/OS.

Use the PROGRAM keyword to define a SIGVER record program name. A single program name is allowed per command.

Note: PROGRAM is also valid as a resource class.

The PROGRAM keyword must be specified when administering the SIGVER record.

The name can contain letters, numbers, or special characters.

Keyword length

One to eight characters.

Capacity of list

One program name per command.

This keyword is used with:

- The commands LIST, ADDTO, REPLACE and REMOVE
- The SIGVER record only
- MISC5(SGVADMIN) authority to maintain the SIGVER record.
- MISC5(SGVLIST) authority to LIST the SIGVER record.

Examples: PROGRAM keyword

This example adds program PROG01 to the SIGVER Record:

```
TSS ADDTO(SIGVER) PROGRAM(prog01)
```

This example removes program PROG01 from the SIGVER Record:

```
TSS REM(SIGVER) PROGRAM(prog01)
```

This example lists the SIGVER entry for program PROG01:

```
TSS LIST(SIGVER) PROGRAM(prog01)
```

PRXBINDDN Keyword—LDAP Server Distinguished Name

Valid on z/OS.

Use the PRXBINDDN keyword as the distinguished name to use when authenticating to the LDAP server. The field can be mixed case and up to 1023 characters in length.

This keyword has the following format:

```
TSS ADDTO|REMOVE(SDT) EIMPROF(eim_profile_name)
                      EIMOPTION(ON|OFF)
                      EIMDOMAIN(name)
                      EIMLOCREG(name)
                      PRXLDAPHST(name)
                      PRXBINDDN(name)
                      PRXBINDPW(password)
```

This keyword is used with:

- The commands ADDTO, REMOVE and REPLACE
- The EIMPROF, EIMOPTION, EIMDOMAIN, EIMLOCREG, PRXLDAPHST, and PRXBINDPW keywords
- The SDT Record exclusively
- MISC(SDT) authority

PRXBINDPW Keyword—Authenticate LDAP Server

Valid on z/OS.

Use the PRXBINDPW keyword to authenticate to the LDAP server. The field can be mixed case and up to 128 characters in length.

This keyword has the following format:

```
TSS ADDTO|REMOVE(SDT) EIMPROF(eim_profile_name)
                      EIMOPTION(ON|OFF)
                      EIMDOMAIN(name)
                      EIMLOCREG(name)
                      PRXLDAPHST(name)
                      PRXBINDDN(name)
                      PRXBINDPW(password)
```

PRXBINDPW is used in conjunction with:

- The commands ADDTO, REMOVE, and REPLACE
- The EIMPROF, EIMOPTION, EIMDOMAIN, EIMLOCREG, PRXLDAPHST, and PRXBINDDN keywords in creating an EIM profile on the SDT
- The SDT Record exclusively
- MISC(SDT) authority

PRXKRBREG Keyword—Kerb Registry Name

Valid on z/OS.

The PRXKRBREG keyword keyword is the name of the Kerb registry. The field can be mixed case and up to 255 characters in length.

For information, see the EIM and PROXY keywords.

When used with ADD or REPLACE, this keyword has the following format:

```
TSS ADDTO(SDT) EIMPROF(eim–profile–name)
               EIMOPTION(ON|OFF)
               EIMDOMAIN(name)
               EIMLOCREG(name)
               PRXLDAPHST(name)
               PRXKRBREG(name)
               PRX509KRB(name)
```

When used with DELETE, this keyword has the following format:

```
TSS DELETE(SDT) EIMPROF(eim–profile–name)
```

This keyword is used with:

- The EIMPROF, EIMOPTION, EIMDOMAIN, EIMLOCREG, PRXLDAPHST, PRXBINDDN, PRXBINDPW, and PRX509KRB keywords
- The SDT Record exclusively
- MISC3(SDT) authority

Example: PRXKRBREG keyword

This example creates an EIMPROF record and add it to the SDT:

```
TSS ADD(SDT) EIMPROF(eim1)
             EIMOPTION(on)
             EIMDOMAIN('xxx–eimDomainName=EIM Domain,
                       o=Company,st=State,c=Country')
             EIMLOCREG('REGISTRY XE')
             PRXLDAPHST('LDAP HOST')
             PRXBINDDN('ldap://domain')
             PRXKRBREG('REGISTRY XE)
             PRX509REG('REGISTRY XE')
             PRXBINDPW(bind password)
```

PRXLDAPHST Keyword—LDAP Server URL and Port

Valid on z/OS.

PRXLDAPHST is the LDAP server and URL and port. The field can be mixed case and up to 1023 characters in length.

This keyword has the following format:

```
TSS ADDTO|REMOVE(SDT) EIMPROF(eim_profile_name)
                      EIMOPTION(ON|OFF)
                      EIMDOMAIN(name)
                      EIMLOCREG(name)
                      PRXLDAPHST(name)
                      PRXBINDDN(name)
                      PRXBINDPW(password)
```

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The keywords EIMPROF, EIMOPTION, EIMDOMAIN, EIMLOCREG, PRXBINDDN, and PRXBINDP
- The SDT Record exclusively
- MISC(SDT) authority

PRX509REG Keyword—Kerb Registry Name

Valid on z/OS.

The PRX509REG keyword is the name of the Kerb registry. The field can be mixed case and up to 255 characters in length.

This keyword has the following format:

Add or Replace EIMPROF record to the SDT:

```
TSS ADDTO|DELETE(SDT) EIMPROF(eim_profile_name)
                      EIMOPTION(ON|OFF)
                      EIMDOMAIN(name)
                      EIMLOCREG(name)
                      PRXLDAPHST(name)
                      PRXKRBREG(name)
                      PRX509REG(name)
```

This keyword is used with:

- The commands ADDTO, REPLACE. and DELETE
- In conjunction with the SDT, EIMPROF, EIMOPTION, EIMDOMAIN, EIMLOCREG, PRXLDAPHST, PRXBINDDN, PRXBINDPW, and PRXKRBREG keywords
- The SDT Record exclusively
- MISC3(SDT) authority

PSTKAPPL Keyword—Application ID

Valid on z/OS.

Use the PSTKAPPL keyword to define the application ID. Depending on the application, the secured signon function uses a specific method to determine the application ID:

- For CICS, IMS, 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 TSO, the application ID is defined by prefacing the SMF identifier of the system with the characters TSO. For example, TSOXE05, is the application ID for TSO on machine XE05. The SMF system ID can be found in SMFPRMxx member of SYS1.PARMLIB.
- For z/OS 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 SMF identifier of the system with the characters z/OS. For example, OS/390 XE05 is the application ID for all batch jobs on machine XE05.

This keyword has the following format:

```
TSS ADDTO(NDT) PSTKAPPL(application)
               SESSKEY(abcdef12)
```

SESSKEY

A hexadecimal “password” that is unique to each application assigned as a PassTicket. A SESSKEY is required for each PassTicket. For information on SESSKEY, see the *User Guide*.

Range: 1 to 16 bytes

This keyword is used with:

- The commands ADDTO, REMOVE, LIST, and REPLACE
- The NDT Record only
- MISC1(NDT) authority

Example: PSTKAPPL keyword

This example indicates that the session key for KA180987 is A1B2C3:

```
TSS ADDTO(NDT) PSTKAPPL(KA180987)
               SESSKEY(A1B2C3)
```

PSUSPEND Keyword—Prevent Access after PTHRESH Violation

Valid on z/OS, z/VSE, and z/VM.

Use the PSUSPEND keyword to prevent ACIDs from accessing the system when a violation occurs due to PTHRESH. Only the the MSCA can issue a command with this keyword.

This keyword has the following format:

```
TSS REMOVE(acid) PSUSPEND
```

This keyword is used with:

- The REMOVE command
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC1(SUSPEND) authority
- Administrators who do not have MISC1(SUSPEND) authority can remove a PSUSPEND from users within their scope if given UPDATE access to entity TSSCMD.USER.REMOVE.PSUSPEND in the CASECAUT resource class

Examples: PSUSPEND keyword

This example removes the PSUSPEND attribute from an ACID:

```
TSS REMOVE(ACARP) PSUSPEND
```

PSWDPHR Keyword—Test Password Phrase

Valid on z/OS 1.8 and above.

Use the PSWDPHR keyword to allow an ACID to sign on with a password phrase when PSWDPHRASE is set OFF.

Note: z/OS 1.10 supports password phrases for TSO signon but changes are required to the IKJTSOxx member in SYS1.PARMLIB. For more information, see IBM's *z/OS MVS Initialization and Tuning Reference*.

For testing purposes, use the PSWDPHR keyword to add password phrases to many ACIDs, but activate those with PSWDPHR only. Then, when satisfied with the PSWDPHR ACID behavior, activate PHRASE for all administered ACIDs by setting PSWDPHRASE to ON.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, and WHOHAS
- ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

This keyword has the format

```
TSS ADDTO(acid) PSWDPHR
```

Example: PSWDPHR keyword

This example adds a password phrase to ACID WOODY and gives the ACID the ability to access the phrase even when PSWDPHRASE(OFF) is set:

```
TSS ADD(WOODY) PHRASE('Robin Hood Robin Hood riding thru the glen')
               PSWDPHR
```

RANGE Keyword—Time Interval

Valid on z/OS, z/VSE, and z/VM.

Use the RANGE keyword to:

- Specify one or more time range intervals within a calendar day associated with a TIMEREC label
- Limit the scope of a search for auto-assignment of UID/GID

For information on using RANGE, see the GID or UID keyword.

This keyword has the following format:

```
TSS ADDTO(SDT) TIMEREC(time-name) RANGE(hhmm:hhmm,...)
```

hhmm:hhmm

Specifies a list of one or more entries in the format hhmm:hhmm for the starting and ending times of a range. You can indicate up to 53 ranges per command. End time must be greater than start time.

Each time is denoted in hours (hh) and minutes (mm) based on a 24-hour day. Values for minutes must be 00, 15, 30, or 45, designating 15-minute increments of a range. For example, specifying 00 for minutes includes minutes 00 to 14 in the range. Therefore, RANGE(1200:1300) signifies 12 noon through 1:14 p.m. If you only want noon to 1 p.m., then specify RANGE(1200:1245) to designate those four quarter hours.

A single time range in the list cannot include midnight. For example, to specify 11 p.m. until 1 a.m., enter:

```
RANGE(2300:2345,0000:0045)
not
RANGE(2300:0045)
```

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The SDT TIMEREC Record exclusively
- MISC3(SDT) authority

Example: RANGE keyword

This example adds a time period from 1 p.m. to 5 p.m. to the RANGE field of the TIME record called TEMP1 in the SDT:

```
TSS ADDTO(SDT) TIMEREC(TEMP1)
               RANGE(1300:1645)
```

RCLASS Keyword—Data Classification Resource Class Definition

Valid on z/OS.

Use the RCLASS keyword to define a Data Classification resource class name. This keyword is part the DATACLAS tracking record that helps determine which data (files, data sets, resources) pertains to which regulation. This record resides in the VSAM file.

The name can contain letters, numbers, or special characters.

Keyword length

One to eight characters.

Capacity of list

One data classification resource class per command.

This keyword is used with:

- The commands LIST, ADDTO, and REMOVE
- The DATACLAS record only
- MISC5(DCLADMIN) authority
- MISC5(DCLLIST) authority to LIST data classes in the DATACLAS record

Examples: RCLASS Keyword

This example adds resource class DATASET to the HIPAA DATACLAS record:

```
TSS ADDTO(DATACLAS) DCLASS(HIPAA)
                    RCLASS(DATASET)
                    RESNAME(HIPAA.RECORDS)
```

This example removes data class HIPAA records with resource DATASET resource class from the DATACLAS record:

```
TSS REMOVE(DATACLAS) DCLASS(HIPAA)
                     RCLASS(DATASET)
                     RESNAME(HIPAS.RECORDS)
```

Note: To remove a record, the DCLASS, RCLASS, RESNAME, OWNER1, and OWNER2 fields must match exactly to list output.

This example lists data concerning data class HIPAA and resource class DATASET:

```
TSS LIST(DATACLAS) RCLASS(DATASET)
```

REALM Keyword—Define Realms

Valid on z/OS.

Use the REALM keyword to define the local realm and foreign realms and their trust relationships with each other. Each organization wishing to run a Kerberos server establishes its own *realm.* The name of the realm in which a client (principal) is registered is part of the client's name. The application server can use this name to decide whether to honor an authentication request.

In a foreign realm, the REALMNAME contains the fully qualified name of both servers in the relationship. The realm name uses the following format:

/…/realm_1/KRBTGT/realm_2

Because the local realm is used to generate keys for the local principal, define your local realm to CA Top Secret before you define local principals. You define a local realm by creating an SDT REALM record with the realm name of KERBDFLT.

When used with local realm, this keyword has the following format:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME('kerberos-realm-name')
               MINTKTLF(min-ticket-life)
               MAXTKTLF(max-ticket-life)
               DEFTKTLF(default-ticket-life)
               KERBPASS(kerberos-password)
               CHKADDRS
```

When used with the foreign realm, this keyword has the following format:

```
TSS ADDTO(SDT) REALM(realm-label)
               REALMNAME('fully-qualified-name')
               KERBPASS(PASSWORD)
```

KERBDFLT

Reserved for the local realm SDT record. Specifies the eight-character local realm.

realm-label

Specifies the identity of the SDT REALM record for foreign realms. The name must be unique and can be up to eight alphanumeric characters. Any REALM name that is not KERBDFLT is assumed to be a foreign realm.

This keyword is used with:

- The commands ADDTO, REMOVE, REPLACE, and LIST
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- MISC3(SDT) authority
- MISC8(LISTSDT) authority to list REALM in the SDT

Examples: REALM keyword

This example creates the default realm record:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME(LOCAL.CA.COM)
               MINTKTLF(30)
               MAXTKTLF(86400)
               DEFTKTLF(36000)
               KERBPASS(CHILDREN)
               CHKADDRS
```

This example deletes the default realm record:

```
TSS REMOVE(SDT) REALM(KERBDFLT)
                CHKADDRS
```

This example lists all REALM records in the SDT:

```
TSS LIST(SDT) REALM(ALL)
```

REALM(FOREIGN_REALM) Keyword—Foreign Node Label

Valid on z/OS.

Use the REALM(FOREIGN_REALM) keyword to provides a label for nodes other than for the local TCP/IP node. Parameters of the REALM must correspond to those of the USS configuration file described by the IBM *Administration Guide* for this service. The label must begin with an alphabetic or national character and may consist of up to 8 alphanumeric characters. The label for a REALM, other than KERBDFLT, is knows as a foreign realm. The REALMNAME specification is used to describe the link between the local URL with the foreign URL.

KERBDFLT is known as the "local" realm.

This keyword has the following format:

```
TSS ADD(SDT) REALM(foreign_realm)
             REALMNAME(fully_qualified_kerberos_name)
             ENCRYPT('[DES|NODES]
                     [DES3|NODES3]
                     [DESD|NODESD]
                     [AES128|NOAES128]
                     [AES256|NOAES256]')
             KERBPASS(password)
```

This keyword is used with:

- The commands ADDTO, REMOVE, REPLACE, and LIST
- The SDT Record exclusively
- MISC3(SDT) authority, granted by the TSS ADMIN function. MISC8(LISTSDT) authority only gives the administrator the ability to list realms in the SDT.

Example: REALM keyword

This example describes the link between the local REALM at HYPOTHETICAL.CA.COM with foreign realm CLIENT1 at HONEYPOT.CLIENT1.COM:

```
TSS ADD(SDT) REALM(CLIENT1)
             REALMNAME(/.../HYPOTHETICAL.CA.COM/krbtgt/HONEYPOT.CLIENT1.COM)
             ENCRYPT('DES DES3')
             KERBPASS(KRYPTO)
```

The ellipsis ("...") in the REALMNAME is a literal part of the required Kerberos naming convention. For information, see the Secureway Server Network Authentication Service documentation.

REALM(KERBDFLT) Keyword—Define Local Realm

Valid on z/OS.

Use the REALM(KERBDFLT) keyword to name the REALM associated with the local TCP/IP node. The definition of the KERBDFLT REALM is required in order to initiate the IBM Secure Way Server Network Authentication Service. The parameters of the REALM must correspond to those of the USS configuration file described by the IBM Administration Guide for this service.

KERBDFLT is known as the “local” realm.

This keyword has the following format:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME(local–realm–URL)
               ENCRYPT('[DES|NODES]
                       [DES3|NODES3]
                       [DESD|NODESD]
                       [AES128|NOAES128]
                       [AES256|NOAES256])
               KERBPASS(password)
               CHKADDRS
```

This keyword is used with:

- The commands ADDTO, REMOVE, REPLACE, and LIST
- The SDT Record exclusively
- MISC3(SDT) authority to ADDTO or REMOVE records or REPLACE fields in the SDT
- MISC8(LISTSDT) authority to list realms in the SDT

Example: REALM(KERBDFLT) Keyword

This example creates the local realm associated with the URL HYPOTHETICAL.CA..COM enabling all encryption types and a password of “THET1CL”

```
TSS ADD(SDT) REALM(KERBDFLT)
             REALMNAME(HYPOTHETICAL.CA.COM)
             ENCRYPT('DES DES3 DESD')
             KERBPASS(THET1CAL)
             CHKADDRS
```

REALMNAME Keyword—Secureway Realm

Valid on z/OS.

Use the REALMNAME keyword to describe the relationship of a REALM in the Secureway Security Server Network Authentication Service configuration file to the local REALM. When REALMNAME is specified for the local REALM, only the URL associated with the local TCP/IP node is employed. When REALMNAME is specified for a foreign realm, a more involved syntax is required.

Note: The format of the REALMNAME supplied in the command is not checked by CA Top Secret.

When used in the local realm, this keyword has the following format:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME(local–realm–URL)
               ENCRYPT('[DES|NODES]
                       [DES3|NODES3]
                       [DESD|NODESD]
                       [AES128|NOAES128]
                       [AES256|NOAES256]')
               KERBPASS(password)
               CHKADDRS
```

When used in the foreign realm, this keyword has the following format:

```
TSS ADDTO(SDT) REALM(foreign_realm)
               REALMNAME('/.../local_URL/krbtgt/foreign_URL')
               ENCRYPT('[DES|NODES]
                       [DES3|NODES3]
                       [DESD|NODESD]
                       [AES128|NOAES128]
                       [AES256|NOAES256]')
               KERBPASS(password)
```

This keyword is used with:

- The commands ADDTO and REPLACE
- The SDT Record exclusively
- MISC3(SDT) authority to ADDTO or REMOVE records or REPLACE fields in the SDT
- MISC8(LISTSDT) authority to list realms in the SDT

REALMNAME(Local Realm) Keyword—Local Realm Name

Valid on z/OS.

Use the REALMNAME keyword to specify the fully qualified names of a local realm name.

This keyword has the following format:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME('kerberos-realm-name')
               MINTKTLF(max-ticket-life)
               MAXTKTLF(max-ticket-life)
               DEFTKTLF(default-ticket-life)
               KERBPASS(kerberos-password)
               CHKADDRS
```

kerberos-realm-name

Specifies the unqualified name of the local realm.

You can use any character except the (X'61') character. Do not use any of the EBCDIC variant characters to avoid problems with different code pages.

Use the single quotes if:

- Parentheses, commas, blanks, or semicolons are entered as part of the name, the character string must be enclosed in single quotes.
- A single quote is part of the name and the entire character string is enclosed in single quotes, use two single quotes together to represent each single quote with the string.
- The first character of the name is a single quote. Enter the string within single quotes, with two single quotes entered for the single quote.

Regardless of the case used, CA Top Secret rolls the name of the local Network Authentication and Privacy Service realm to upper case. This does not ensure that a valid *kerberos-realm-name* has been specified.

Maximum length: 117 characters

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE LIST
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- MISC3(SDT) authority
- MISC8(LISTSDT) authority to list KERBNAME in the SDT

Example: REALMNAME keyword

This example creates the default local realm record with a realm name of Local.ca.com:

```
TSS ADDTO(SDT) REALM(KERBDFLT)
               REALMNAME(LOCAL.CA.COM)
               MINTKTLF(30)
               MAXTKTLF(86400)
               DEFTKTLF(36000)
               KERBPASS(CHILDREN)
               CHKADDRS
```

REALMNAME Keyword—Specify Local and Foreign Realm Name

Valid on z/OS.

Use the REALMNAME keyword to specify the fully qualified names of both the local realm name and the foreign realm name in a trust relationship.

This keyword has the following format:

```
TSS ADDTO(SDT) REALM(realm-label)
               REALMNAME('fully-qualified-name')
               KERBPASS(password)
```

fully-qualified-name

Specifies the fully qualified name of the foreign realm.

Can be any character except the (X'61') character. Do *not* use any of the EBCDIC variant characters to avoid problems with different code pages.

Use the single quotes if:

- Parentheses, commas, blanks, or semicolons are entered as part of the name, the character string must be enclosed in single quotes.
- A single quote is part of the name and the entire character string is enclosed in single quotes, use two single quotes together to represent each single quote with the string.
- The first character of the name is a single quote. Enter the string within single quotes, with two single quotes entered for the single quote.

Regardless of the case used, CA Top Secret rolls the name to upper case. This does not ensure that a valid REALMNAME has been specified.

Maximum length: 240 characters

This keyword is used with:

- The commands ADDTO, REMOVE, REPLACE, and LIST
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- MISC3(SDT) authority
- MISC8(LISTSDT) authority to list KERBNAME in the SDT

Example: REALMNAME keyword

This example creates a foreign realm SDT record with a principal name of LOCAL.CA.COM and a realm password of children:

```
TSS ADDTO(SDT) REALM(KERBFOR1)
               REALMNAME('LOCAL.CA.COM/KRBTGT/FOREIGN.SERVER_1')
               KERBPASS(CHILDREN)
```

RECDATA Keyword—RECORD Field Characteristics

Valid on z/OS, z/VSE, and z/VM.

Use the RECDATA keyword to specify field characteristics associated with a RECORD. Administrators can add, remove, or replace a RECDATA field in the RLP record of the SDT Record.

This keyword has the following format:

```
TSS ADDTO(SDT) RECORD(rlp-name)
               RECDATA(field-definition)
```

field-definition

Contains the layout of the field to be masked. You can specify up to five fields on each RLP keyword. Each field-definition operand consists of four sub-fields:

fld-name

Specifies a field name unique to this RLP record. The name can consist of letters, digits, or national characters and can be qualified with periods (.), underscores (_), or hyphens (-).

Size: Up to 24 characters

type

Indicates the field type. Replace type with one of these values:

- CHAR. Specifies any of the EBCDIC characters, left-justified and padded with blanks.
- BIN. Specifies up to 16 decimal digits with optional sign. If no sign is specified, a positive value is assumed.
- PACKED. Specifies up to 16 decimal digits with optional sign. If no sign is specified, a positive value is assumed.
- ZONED. Specifies up to 16 decimal digits with optional sign. If no sign is specified, a positive value is assumed.
- HEX. Specifies up to 256 hexadecimal digits (0-F) left-justified and padded with nulls (X"00").

offset

Specifies a five-digit initial position (relative to 1) of the data field.

length

(Optional) Specifies the length of the data field. If the length field is omitted, it indicates a length large enough to include the rest of the record.

Size: Up to 44 characters

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The SDT Record exclusively
- MISC3(SDT) authority

Examples: RECDATA keyword

This example adds a RECDATA field to the RLP record PROLL1 that contains a name field in character format called NAME, having an offset of 20 and a length of 30 characters:

```
TSS ADDTO(SDT) RECORD(PROLL1)
               RECDATA(NAME,CHAR,20,30)
```

This example replaces an existing record with one new field

```
TSS REPLACE(SDT) RECORD(PROLL1)
                 RECDATA(NAME,CHAR,20,15).
```

Even though only one field's worth of data is entered with the REPLACE syntax, all previously entered fields are replaced by the single field entered in the REPLACE command. If the RLP record consists of seven fields, all seven original fields are lost and one new field replaces them. There is no command to replace a single field.

RECORD Keyword—RLP Record Name

Valid on z/OS, z/VSE, and z/VM.

Use the RECORD keyword to provide an up to eight-character name associated with each RLP Record in the SDT Record. RLP records are associated with CICS FCT entries by name.

This keyword has the following format:

```
TSS ADDTO(SDT) RECORD(rlp-name)
               RECDATA(field-definition1,...field-definition5)
```

rlp-name

Specifies a user-defined record ID that must be unique for each RLP record that can contain letters, numbers, and special characters. The specified rlp-name must match the name of the FCT being protected by the RLP feature.

Size: Up to 8 characters

This keyword is used with:

- The commands ADDTO, DELETE, REMOVE, REPLACE, and LIST
- The SDT Record exclusively
- MISC3(SDT) authority

Example: RECORD Keyword

This example creates 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:

```
TSS ADDTO(SDT) RECORD(PROLL1)
               RECDATA(PAY,CHAR,48,6)
```

RESCLASS Keyword—Resource Classes

Valid on z/OS, z/VSE, and z/VM.

Use the RESCLASS keyword to define a resource-class-name that allows one resource-class-name per command. The CA Top Secret command, logging, and the security interface honor this name.

The name can contain letters, numbers, or special characters.

To avoid a dynamically-defined resource conflicting with a predefined CA Top Secret resource class, give the dynamically-defined resource class a national (@,#,$) or numeric (0-9) in one of the first four characters of the name.

The first four characters of all resource class names must be unique and cannot conflict with any dynamically-defined or predefined resource class name. For example, VOLUME is an existing predefined RESCLASS name; therefore, it is not possible to create a new RESCLASS name called VOL.

The resource class is immediately usable with all RACF macros. For new resource classes, there is a limitation of 10 that can be added between IPLs for purposes of RACROUTE REQUEST=STAT or (RACSTAT). Any resource classes, beyond the limit of 10, that are added between IPLs are not found by RACSTAT until after the next IPL.

RESCLASS(ALT-ACID) for a TSS LIST command is not valid. You cannot show cross-authorization for other acids.

There are 126 user-defined resource classes available in the RDT.

This keyword has the following format:

```
TSS ADDTO(RDT) RESCLASS(resource class)
               RESCODE(hex code)
```

Keyword length

One to eight characters.

Capacity of list

One resource class per command: RESCLASS and RESCODE are required when adding a resource to the RDT Record

This keyword is used with:

- The commands REPLACE, LIST, ADDTO, and REMOVE
- The RDT record only
- MISC1(RDT) authority
- MISC1(RDT) or MISC8(LISTRDT) authority to LIST resource classes in the RDT Record

Examples: RESCLASS keyword

This example adds #PRODUCT to the RDT Record:

```
TSS ADDTO(RDT) RESCLASS(#PRODUCT)
               RESCODE(13F)
```

This example removes #PRODUCT from the RDT Record:

```
TSS REMOVE(RDT) RESCLASS(#PRODUCT)
```

Note: Removing a resource class from the RDT Record requires that all ownership of the resource class is previously removed.

This example lists data concerning how FCTs are processed:

```
TSS LIST(RDT) RESCLASS(FCT)
```

resclass(resource) Keyword—Associate Resource with SECLABEL

Valid on z/OS.

Use the resclass(resource) keyword to associate a resource instance with a SECLABEL and optionally override MLMODE for matching resources.

This keyword has the following format:

```
TSS ADD(MLS) resclass(resource.instance)
             SECLABEL(label)
             [MODE(FAIL|WARN|DORM)]
```

resclass

Defines an RDT resource class. It is not necessary to define the entire RESCLASS to MLS in order to define a single instance.

resource.instance

Defines a resource instance within the specified resource class already defined to CA Top Secret. The resource instance may be masked.

label

Defines a security label defined to the MLS record.

mode

Defines the mode MLS security interprets MLS security for matching resources of this resource instance. Optionally overrides MLMODE.

This keyword is used the commands ADDTO and REMOVE.

Example: resource keyword

This example associates a resource instance with a SECLABEL.

```
TSS ADD(MLS) OTRAN(PAYR)
             SECLABEL(LABELAP)

TSS ADD(MLS) SERVAUTH(ezb.netaccess.—.zone2) seclabel(label2)

TSS ADD(MLS) DB2(TEST.QEWRQER.*.ASDF)
             SECLABEL(LABELA)
```

RESCODE Keyword—Resource Class Abbreviation

Valid on z/OS, z/VSE, and z/VM.

Use the RESCODE keyword to:

- Supply an internal abbreviation for the RESCLASS resource class in ACID security records and in audit information
- To find what RESCLASS is associated with a specific resource code

This keyword has the following format:

```
TSS ADDTO(RDT) RESCLASS(resource Type)
               [RESCODE(hex code)]
```

RESCODE

Values have assigned meanings:

001 - 03F

Available for user-defined RIE resources.

101 - 13F

Available for user-defined PIE resources.

All others

Reserved for CA Top Secret use.

Default: The first available unused user-definable value, either RIE or PIE. To define a RIE or PIE resource, specify an appropriate RESCODE value.

This keyword is used with:

- The commands ADDTO and LIST
- The RDT ACID only
- MISC1(RDT) authority
- MISC1(RDT) or MISC8(LISTRDT) authority to LIST resource classes in the RDT Record

Examples: RESCODE keyword

This example adds #PRODUCT to the RDT Record:

```
TSS ADDTO(RDT) RESCLASS(#PRODUCT)
               RESCODE(13F)
```

This example removes #PRODUCT from the RDT Record:

```
TSS REMOVE(RDT) RESCLASS(#PRODUCT)
```

This example lists data concerning how resource code 13F is processed:

```
TSS LIST(RDT) RESCODE(13F)
```

This example determines what RESCLASS is using the resource code x'D2':

```
TSS LIST(RDT) RESCODE(D2)
```

RESNAME Keyword—Data Classification Resource Entity

Valid on z/OS.

Use the RESNAME keyword to define a Data Classification resource entity. This keyword is part the DATACLAS tracking record that helps determine which data (files, data sets, resources) pertains to which regulation. This record resides in the VSAM file.

The name can contain letters, numbers, or special characters. To specify that the RESNAME is a prefix, append an asterisk (*). No other masking is supported. Any other use of masking characters will be treated as literals.

Keyword length

One to 255 characters.

Capacity of list

One data classification resource class per command.

This keyword is used with:

- The commands LIST, ADDTO, and REMOVE
- The DATACLAS record only
- MISC5(DCLADMIN) authority
- MISC5(DCLLIST) authority to LIST data classes in the DATACLAS record

Examples: RESNAME Keyword

- This example adds resource class DATASET to the HIPAA DATACLAS record:

```
TSS ADDTO(DATACLAS) DCLASS(HIPAA)
    RCLASS(DATASET)
    RESNAME(HIPAA.RECORDS)
```

- This example adds resource prefix HIPAA.* to the HIPAA DATACLAS record:

```
TSS ADDTO(DATACLAS)
    DCLASS(HIPAA)
    RCLASS(DATASET)
    RESNAME(HIPAA.*)
```

- This example removes data class HIPAA records with resource DATASET resource class from the DATACLAS record:

```
TSS REMOVE(DATACLAS)
    DCLASS(HIPAA)
    RCLASS(DATASET)
    RESNAME(HIPAA.RECORDS)
```

  Note: To remove a record, the DCLASS, RCLASS, RESNAME, OWNER1, and OWNER2 fields must exactly match list output.

RESOURCE Keyword—Global Resource Class Administration Authority

Valid on z/OS, z/VSE, and z/VM.

Use the RESOURCE keyword with the ADMIN command to give authority for an CA Top Secret administrator to issue ADDTO, LIST, REMOVE, PERMIT, REVOKE, and WHOHAS commands for a specific resource class defined in the RDT applied to any ACID owned within its administrative scope. When a resource class contains access levels, the administration can be limited to ACCESS in one or more access-levels; if the administrator is to manipulate all access levels, specify ACCESS(ALL).

Use the RESOURCE keyword with the DEADMIN command to disallow administrative authority of RESOURCE manipulation, ACCESS can be specified by DEADMIN, but it is ignored.

Note: Authority can also be granted/removed to administer all resources globally.

This keyword has the following format:

```
TSS ADMIN(acid) RESOURCE(authority-level(s))
                ACCESS(access-level(s))

TSS DEADMIN(acid) RESOURCE(authority-level(s))
```

This keyword can be used with:

- The commands ADMIN, DEADMIN, LIST and WHOHAS

RESOURCE Authority Levels

The CA Top Secret administrator may specify one or more of the following authority levels:

ALL

Gives the named "ACID" any of the authorities listed above.

AUDIT

Gives the named "acid" the ability to ADDTO or REMOVE any resource prefixes from the Audit Record. For details, see the *Auditor's Guide*.

INFO

Gives the named "ACID" the ability to employ WHOOWNS and WHOHAS for any resource.

OWN

Gives the named "ACID" the administrative authority to ADDTO or REMOVE resources for acids under its scope of control.

REPORT

Gives the named "ACID" the ability to obtain reports for all resources by employing the utilities TSSUTIL, TSSAUDIT, TSSCPR, and TSSCHART.

XAUTH

Gives the named "ACID" the administrative authority to PERMIT or REVOKE resources for acids under its scope of control.

Access Levels

When granting XAUTH authority to the named “ACID,” the administrator may limit the access levels which the named “ACID” can PERMIT.

DEFAULT

If the ADMIN command does not specify an ACCESS clause, the named "ACID" of the command is not allowed to specify an ACCESS keyword in PERMIT commands. As a result, all PERMIT commands issued by the named "ACID" will default to the DEFACC access-level defined in the RDT

ALL

Named "ACID" may permit any resource at any access level.

CONTROL

Named "ACID" may permit any resource at the access level CONTROL.

CREATE

Named "ACID" may permit any resource at the access level CREATE.

DELETE

Named "ACID" may permit any resource at the access level DELETE.

FEOV

Named "ACID" may permit any resource at the access level FEOV.

FETCH

Named "ACID" may permit any resource at the access level FETCH.

NONE

Named "ACID" may permit any resource at the access level NONE.

PURGE

Named "ACID" may permit any resource at the access level PURGE.

READ

Named "ACID" may permit any resource at the access level READ.

REPLACE

Named "ACID" may permit any resource at the access level REPLACE.

SCRATCH

Named "ACID" may permit any resource at the access level SCRATCH.

UPDATE

Named "ACID" may permit any resource at the access level UPDATE.

WRITE

Named "ACID" may permit any resource at the access level WRITE.

Note: The ACLST of the resource class definition in the RDT governs the appropriate use of ACCESS levels in PERMIT commands for individual resources.

More information:

ACLST Keyword—Resource Access Level (see page 114)

Examples: RESOURCE keyword

This example authorizes an administrator to PERMIT users to update any resource owned within his scope, and to determine who owns and who has access to those resources:

```
TSS ADMIN(SUPSCA) RESOURCE(XAUTH,INFO)
                  ACCESS(U)
```

This example removes SUPSCA's authority for resources:

```
TSS DEADMIN(SUPSCA) RESOURCE(XAUTH,INFO)
```

RESOURCE Keyword—Individual Resource Class Administration Authority

Valid on z/OS, z/VSE, and z/VM.

Use the *resource* keyword to grant administrative authority for a specific resource class to an individual administrator.

This keyword has the following format:

```
TSS ADMIN(acid) resource(authority-level(s))
```

resource

The specific resource class RESCLASS from the RDT.

This keyword can be used with:

- The commands ADMIN, DEADMIN, LIST and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA

More information:

RESOURCE Keyword—Global Resource Class Administration Authority (see page 431)

RESOURCE Authority Levels

The CA Top Secret administrator may specify one or more of the following authority levels:

ALL

Gives the named "ACID" any of the authorities listed above.

AUDIT

Gives the named "acid" the ability to ADDTO or REMOVE any resource prefixes from the Audit Record. For details, see the *Auditor's Guide*.

INFO

Gives the named "ACID" the ability to employ WHOOWNS and WHOHAS for any resource.

OWN

Gives the named "ACID" the administrative authority to ADDTO or REMOVE resources for acids under its scope of control.

REPORT

Gives the named "ACID" the ability to obtain reports for all resources by employing the utilities TSSUTIL, TSSAUDIT, TSSCPR, and TSSCHART.

XAUTH

Gives the named "ACID" the administrative authority to PERMIT or REVOKE resources for acids under its scope of control.

Access Levels

When granting XAUTH authority to the named “ACID,” the administrator may limit the access levels which the named “ACID” can PERMIT.

DEFAULT

If the ADMIN command does not specify an ACCESS clause, the named "ACID" of the command is not allowed to specify an ACCESS keyword in PERMIT commands. As a result, all PERMIT commands issued by the named "ACID" will default to the DEFACC access-level defined in the RDT

ALL

Named "ACID" may permit any resource at any access level.

CONTROL

Named "ACID" may permit any resource at the access level CONTROL.

CREATE

Named "ACID" may permit any resource at the access level CREATE.

DELETE

Named "ACID" may permit any resource at the access level DELETE.

FEOV

Named "ACID" may permit any resource at the access level FEOV.

FETCH

Named "ACID" may permit any resource at the access level FETCH.

NONE

Named "ACID" may permit any resource at the access level NONE.

PURGE

Named "ACID" may permit any resource at the access level PURGE.

READ

Named "ACID" may permit any resource at the access level READ.

REPLACE

Named "ACID" may permit any resource at the access level REPLACE.

SCRATCH

Named "ACID" may permit any resource at the access level SCRATCH.

UPDATE

Named "ACID" may permit any resource at the access level UPDATE.

WRITE

Named "ACID" may permit any resource at the access level WRITE.

Note: The ACLST of the resource class definition in the RDT governs the appropriate use of ACCESS levels in PERMIT commands for individual resources.

More information:

ACLST Keyword—Resource Access Level (see page 114)

Example: resource Keyword

This example allows the RESADM ACID to permit READ access to DATASET resources for ACIDs when both the ACID and the DATASET are owned and within scope.

```
TSS ADMIN(RESADM) DATASET(XAUTH)
                  ACCESS(READ)
```

RESOWNER Keyword—Maintain SMS ACID

Valid on z/OS.

Use the RESOWNER keyword to add or remove a User or Control type ACID associated with a data set or an ACID that matches the high level index of the data set being allocated for the extraction by SMS.

The ownership of the data set must be established if it is not already owned by the ACID.

This keyword has the following format:

```
TSS ADDTO(acid) DSNAME(dataset name)
                RESOWNER(smsacid)
```

This keyword is used with:

- The command ADDTO
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- DSNAME(OWN) authority

Examples: RESOWNER keyword

This example adds a RESOWNER of SMSACID to data set PAY.MASTER:

```
TSS ADDTO(PAYDEPT) DSNAME(PAY.MASTER)
                   RESOWNER(SMSACID)
```

This example removes a RESOWNER of SMSACID from data set PAY.MASTER:

```
TSS ADDTO(PAYDEPT) DSNAME(PAY.MASTER)
                   RESOWNER()
```

RETAIN Keyword—Leave Data Set in Hyperspace

Valid on z/OS.

Use the RETAIN keyword with the DLF ACID to leave the data set in hyperspace when the job which bought the data set into hyperspace ends. The RETAIN keyword must be used with the DSNAME resource.

This keyword has the following format:

```
TSS ADDTO(DLF) DSNAME(nnnnn)
               RETAIN
```

This keyword is used with:

- The commands ADDTO and REMOVE
- The DLF ACID only
- MISC2(DLF) authority

Examples: RETAIN keyword

This example adds the RETAIN attribute to an existing DLF authorized data set or adds the data set to the DLF Record with the RETAIN attribute.

```
TSS ADDTO(DLF) DSNAME(CICS.MASTER.FILE)
               RETAIN
```

This example leaves data set authorized for DLF, but removes the RETAIN attribute:

```
TSS REMOVE(DLF) DSNAME(CICS.MASTER.FILE)
                RETAIN
```

RINGDATA Keyword—Add Certificate to a Ring

Valid on z/OS.

Use the RINGDATA to add a digital certificate to a key ring. The keyword RINGDATA specifies the ACID and certificate name (as specified by DIGICERT) of the digital certificate being added to the user.

This keyword has the following format:

```
TSS ADDTO(userid) KEYRING(8-byte name)
                  [LABLRING(237-byte ring name)]
                  {RINGDATA(userid,digicert)}
                  {RINGDATA(CERTSITE,digicert)}
                  {RINGDATA(CERTAUTH,digicert)}
                  [DEFAULT]
                  [USAGE(PERSONAL|CERTSITE|CERTAUTH)]
```

userid,digicert

ACID and name of digital certificate

CERTSITE,*digicert*

This certificate is a site-generated certificate

CERTAUTH,*digicert*

This certificate is from a certificate authority.

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTUSER) authority for user IDs
- MISC4(CERTSITE) for CERTSITE ACIDs
- MISC4(CERTAUTH) for CERTAUTH ACIDs

Examples: RINGDATA keyword

This example specifies the ACID and certificate name of the certificate being added to the user key ring:

```
TSS ADDTO(user01) KEYRING(CASRING)
                  RINGDATA(user01,cert0001)
```

This example removes a certificate in a key ring:

```
TSS REMOVE(user01) KEYRING(CASRING)
                   RINGDATA(user01,cert0001)
```

RSTDACC keyword—Control Access to Resources

Valid on z/OS, z/VSE, and z/VM

Use the RSTDACC keyword to prevent CA Top Secret users from accessing protected resources.

The RSTDACC keyword is effective only under USS and only when control option HFSSEC is set to OFF.

Access is allowed:

- If the file USER bit of the accessing user matches the file owner UID
- If the GROUP bits of the file owning group match one of the accessing user's GIDs.
- If the file has an associated ACL entries which allow additional UID and GID specifications

Access may also be allowed based on the OTHER bit default access if the user has the RSTDACC keyword and READ access to:

```
UNIXPRIV(RESTRICTED.FILESYS.ACCESS)
```

The following table presents READ access scenarios:

User is "restricted"	READ access to UNIXPRIV(RESTRICTED.FILESYS.ACCESS)	Result
Yes	Yes	Check "other" bits to determine access.
Yes	No	Bypass check of "other" bits and deny access.

No	Yes	Check "other" bits to determine access.
No	No	Check "other" bits to determine access.

This keyword has the following format:

```
TSS ADDTO(acid) RSTDACC
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA,and MSCA
- MISC7(RSTDACC) and ACID(MAINTAIN) authority

Example: RSTDACC keyword

This example allows public to bypass security checks for resource access (except data sets and volumes):

```
TSS ADDTO(public) RSTDACC
```

This example removes the RSTDACC attribute:

```
TSS REMOVE(public) RSTDACC
```

SCOPE Keyword—Allow Authority

Valid on z/OS, z/VSE, and z/VM.

Use the SCOPE keyword for the MSCA to allow authority for:

```
TSS ADMIN(lsca) SCOPE({lsca1}[,{lsca2},...]) {zone1} {zone2}
```

```
TSS DEADMIN(dca1|vca1|zca1|lsca1) SCOPE(profile1)
```

Note: The MSCA is the only one able to grant an administrative acid scope over a profile not in their normal scope.

This keyword has the following format:

```
TSS ADMIN(lsca) SCOPE(lsca1|zone1,...)
```

```
TSS ADMIN(admin) SCOPE(profile)
```

This keyword can be used with the commands ADMIN and DEADMIN. You can also use this keyword for the WHOHAS command when used with AUTHADM.

Only the MSCA can assign:

- An LSCA's SCOPE. This allows the LSCA to administer all of the ACIDs in their scope within the bounds of their administrative authority.
- An administrative ACID scope over a profile. This allows the administrative ACID to ADD or REMOVE the specified PROFILE to ACIDs It does not allow the administrative ACID to manipulate objects inside the profile. The administrative acid cannot ADD or REMOVE keywords from the PROFILE the administrative acid cannot PERMIT or REVOKE resources from the PROFILE.

Note: Administration of profiles may not be given to ACIDs of type USER or LSCA.

Examples: SCOPE keyword

This example gives LSCA01 administrative authority over LSCA02 and ZONE1:

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

To remove an ACID or ACIDs from the LSCA's scope, use the DEADMIN command function and specify each ACID being removed.

This example removes all lower level administrative authorities from LSCA01:

```
TSS DEADMIN(LSCA01) SCOPE(LSCA02,ZONE1)
```

This example gives an administrator an administrative ACID scope over a profile not normally within the ACID's scope of authority. In this example, DCA2 administers DEPT2, and PROF1 resides in DEPT1.

This example gives DCA2 administrative authority over PROF1:

```
TSS ADMIN(DCA2) SCOPE(PROF1)
```

This example grants DCA1 scope over a profile PROFILE1:

```
TSS ADMIN(dca1) SCOPE(profile1)
```

This example allows DCA1 to add and remove the profile from acids in their scope:

```
TSS ADD(user1) PROFILE(profile1)
```

SCTYKEY Keyword—Specify CICS Security Keys Use

Valid on z/OS and z/ VSE.

Use the SCTYKEY keyword to specify which CICS security keys an ACID may or may not use.

This keyword has the following format:

```
TSS ADDTO(acid) SCTYKEY(n,n,...)
```

Capacity of list

Up to 256 CICS security keys per command. (CICS only recognizes up to 64 security keys per TSS command.)

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Security keys are available in all modes except FAIL.

Examples: SCTYKEY keyword

This example specifies security keys for user VCA14:

```
TSS ADDTO(VCA14) SCTYKEY(1,5,7,11)
```

This example removes security keys:

```
TSS REMOVE(VCA14) SCTYKEY(1,5,7)
```

SDNFILTR Keyword—Portion of DN for Filter

Valid on z/OS.

Use the SDNFILTR keyword to specify the significant portion of the subject's distinguished name used as a filter when associating an ACID with a digital certificate on a TSS ADD CERTMAP command. The SDNFILTR data must match a portion of the subject's distinguished name extracted from the certificate. The distinguished name from the point of the match to the end of the name is used as the filter data.

When specified with DCDSN, the filter must correspond to a starting point within the subject's distinguished name found in the certificate contained in the data set. Specify enough of the name to precisely identify the starting point for the filter.

For example, if the certificate in the data set has CN=Bob Smith.OU=BobsAccountingDept.O=BobsMart.L=internet as a subject and you want all certificates for anyone in Bob's Accounting Department selected by this filter, specify:

```
SDNFILTR('OU=BobsAcc')
```

Without the data set containing the certificate, enter the following to produce the same result:

```
SNFILTR('OU=BobsAccountingDept.O=BobsMart.L=internet')
```

SDNFILTR is optional if the issuer's distinguished name filter (IDNFILTR) is specified. If SDNFILTR is not specified, only the issuer's name is used as a filter. SDNFILTR must not be specified with IDNFILTR unless the value of IDNFILTR will result in the entire issuer's name being used in the filter. Note that the subject's name can be partial but cannot be used in a filter that contains only a partial issuer's name.

A maximum of 255 characters can be entered for SDNFILTR if SDNSIZE(255) control option is specified. A maximum of 1024 characters can be entered for SDNFILTR if SDNSIZE(1024) control option is specified. When a starting value is specified for a certificate contained in a data set, there cannot be more than 255/1024 characters between the starting point and the end of the issuer's name in the certificate.

This keyword has the following format:

The SDNFILTR value must be enclosed in quotes.

```
TSS ADDTO(acid) CERTMAP(recid)
                SDNFILTR('subject-dist-name-filter')
```

'*subject-dist-name-filter*'

Specifies the significant portion of the subject's distinguished name to be used as a filter. The value specified for SDNFILTR must begin with a prefix found in the following list, followed by an equal sign (X'7E'). Each component should be separated by a period (X'4B'). The case, blanks, and punctuation displayed when the digital certificate information is listed must be maintained in the SDNFILTR. Since digital certificates only contain characters available in the ASCII character set, the same characters should be used for the SDNFILTR value. Valid prefixes are:

- COUNTRY (specified as C=)
- STATE/PROVINCE (specified as ST=)
- LOCALITY (specified as L=)
- ORGANIZATION (specified as O=)
- ORGANIZATIONAL UNIT (specified as OU=)
- TITLE (specified as T=)
- COMMON NAME (specified as CN=)
- DOMAIN COMPONENT (specified as DC=)
- POSTAL CODE (specified as PC=)
- EMAIL (specified as E=)
- STREET NAME (specified as STREET=)
- USERID (specified as UID=)

This keyword is used with:

- The commands ADDTO, LIST, and REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Example: SDNFILTR keyword

This example assigns users who enter the system with a certificate subject of OU=Dept3.OU=NY.OU=Sales.O=ABC Co acid NYDEPT3.

```
TSS ADDTO(NJDEPT3) CERTMAP(NJMAP3)
                   LABLCMAP('NY Dept 3 Map')
                   TRUST
                   SDNFILTR('OU=Dept3.OU=NY.OU=Sales.O=ABC Co')
```

SDTFNAME Keyword—Remove SDT Record Field

Valid on z/OS, z/VSE, and z/VM.

Use the SDTFNAME keyword to remove a field from a MASK, MAP or RECORD in the SDT Record.

This keyword has the following format:

```
TSS REMOVE(SDT) [MASKREC(mask-name)]
                SDTFNAME(data-fld1,...data-fld5)
                [MAPREC(map-name)]
                RECORD(rlp-name)
```

data-fld

Any data field associated with the MAPDATA, MASKDATA, RECDATA and SELDATA fields.

This keyword is used with:

- The REMOVE command
- The SDT Record exclusively
- MISC3(SDT) authority

Example: REMOVE keyword

This example removes a MASKDATA field called MASK1 from the PAYMASK SDT MASK record:

```
TSS REMOVE(SDT) MASKREC(PAYMASK)
                SDTFNAME(MASK1)
```

SECLABEL Keyword—Security Labels

Valid on z/OS.

Use the SECLABEL keyword to define and remove security labels in the MLS record. A security administrator can create the security labels assigned to users, data, and resources. A list of available SECLABEL names can be assigned to an ACID, as well as a default label DFLTSLBL.

This keyword has the following format:

```
TSS ADDTO(MLS) SECLABEL(seclabel–name)
               SECEVEL(level–name)
              CATEGORY(category1,…,categoryn)
               SYSID(sysid1,…,sysidn)
               MLAUDIT(access1,…,accessn)

TSS REMOVE(MLS) SECLABEL(seclabel–name)
```

SECLABEL

Specifies the alphanumeric name of a security label. A SECLABEL cannot be removed from MLS if it is associated with a user or a resource class.

Size: 8 bytes

LEVEL

Specifies the name of a security level that was defined in the MLS record.

CATEGORY

(Optional) Specifies one or more categories that were defined in the MLS record.

SYSID

(Optional) Specifies one or more system IDs on which this security label can be used.

MLAUDIT(access[,access2,...,access8])

Specifies a list of up to eight access levels, under which MLS auditing is initiated for objects with the selected SECLABEL. Valid access types are READ, CREATE, WRITE, CONTROL, UPDATE, SCRATCH, FETCH, ALTER, and ALL. To audit the seclabel without regard to access, use ACCESS(ALL). If any other access level is entered or if MLAUDIT is not specified the default value READ is substituted.

The MLSECAUD control option must be activated for seclabel auditing to be performed. For information, see the *Control Options Guide*.

MLS seclabel auditing supports the ck_access, ck_process_owner, ck_owner_two_files, and R_ptrace UNIX callable services. Valid access levels to perform MLS seclabel auditing are:

- READ—ck_access
- WRITE—ck_access
- CONTROL—ck_process_owner, ck_owner_two_files, and R_ptrace

The corresponding UNIX access type is displayed when reporting ck_access.

This keyword is used with:

- The commands ADDTO, LIST, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC5(MLS) authority

Examples: SECLABEL keyword

This example adds SECLABEL ADMIN to the MLS Record:

```
TSS ADDTO(MLS) SECLABEL(ADMIN)
               CATEGORY(PERSONNEL,ACCOUNTING)
                SECLEVEL(10)
                SYSID(SYS1)
```

This example removes the SECLABEL ADMIN:

```
TSS REMOVE(MLS) SECLABEL(ADMIN)
```

This example adds seclabel ADMIN to the dataset PAYROLL.RECORDS:

```
TSS ADDTO(MLS) DSN(PAYROLL.RECORDS)
               SECLABEL(ADMIN)
```

SECLEVEL Keyword—Security Level

Valid on z/OS.

Use the SECLEVEL keyword to define or remove security levels, which are the hierarchical elements of security labels.

This keyword has the following format:

```
TSS ADDTO(MLS) SECLEVEL(level–number)
               LVLNAME(level–name)

TSS REMOVE(MLS)SECLEVEL(level–name)
```

SECLEVEL

(Required) Specifies the unique internal numeric value of a level. Higher numbers represent higher levels.

Range: 1 to 254

LVLNAME

Specifies the alphanumeric name of a security level.

Range: 1 to 255

This keyword is used with:

- The commands ADDTO, LIST, REMOVE, and REPLACE
- MISC5(MLS) authority

Example: Add Security Level 10 to the MLS Record

This example adds security level 10 to the MLS Record:

```
TSS ADDTO(MLS) SECLEVEL(10)
               LVLNAME('Department Administrator')
```

Example: Remove Security Level 10 from the MLS Record

This example removes security level 10:

```
TSS REMOVE(MLS) SECLEVEL(10)
```

SEGMENT Keyword—Field Segments

Valid on z/OS, z/VSE, and z/VM.

Use the SEGMENT keyword to perform the following tasks:

- Assign fields to a specific segment. You cannot add user-defined fields to system-defined segments.
- List data about fields in a specific segment.

This keyword has the following format:

```
TSS ADDTO|LIST(FDT) SEGMENT(segmentname)
```

segmentname

Specifies the name of the segment.

Range: 1 to 8

Default: BASE

The pre-defined CA Top Secret segments are as follows:

- ALL
- BASE
- CICS
- DCE
- DFP
- DLFDATA
- IMAPDATA
- IMS
- LANGUAGE
- NETVIEW
- OPERPARM
- SESSION
- TSO
- WORKATTR
- z/OS

This keyword is used with:

- The commands ADDTO, LIST, REMOVE, and REPLACE
- The ACID types ACID, FDT, and USER

- MISC1(RDT) authority

Examples: SEGMENT Keyword

This example includes a field called $ACCTS in a segment named TAXINFO:

```
TSS ADDTO(FDT) FDTNAME($ACCTS)
               FDTCODE(1E)
               SEGMENT(TAXINFO)
               MAXLEN(12)
               DISPLAY('SALARY ACCT')
```

This example finds all field names in the TAXINFO segment:

```
TSS LIST(FDT) SEGMENT(TAXINFO)
```

This example lists all fields associated with the specified SEGMENT:

```
TSS LIST(JDoe) SEGMENT(IMAPDATA)
```

SELDATA Keyword—SDT SELDATA Field

Valid on z/OS and z/VSE.

Use the SELDATA keyword to add, remove, or replace a SELDATA field in the SELECT record of the SDT Record. SELDATA allows the administrator to specify a (compound) logical condition to decide whether RLP or Screen Level Protection is applicable for an associated SELECT.

This keyword has the following format:

```
TSS ADDTO(SDT) SELECT(sel-name)
               DESCRIPT(desc-name)
               SELDATA('IF [NOT] sel-expression AND|OR] sel-expression')
```

IF

Signals the beginning of a select expression. Specify IF, and the entire SELDATA must be enclosed in a single quote.

NOT

Specifies a negative relationship for the following sel-expression.

sel-expression

Specifies the logic behind which records the user is restricted to. The sel-expression is coded in a Boolean logic format, which must contain a left-hand side (LHS) field name, a relational operator, and a right-hand side (RHS) comparison data value.

A sel-expression consists of Boolean expressions modified by NOT, surrounded by parentheses, and joined by the conjunctions AND and OR. Parentheses are used to determine the order of evaluation. There is no priority applied to AND, OR, and NOT when parentheses are not supplied. The expression consists of:

- left-hand side—Specifies the field name (as defined in the SDT RECORD) that you want CA Top Secret to compare. If the field name is not defined to CA Top Secret via the SDT RECORD, access to the requested record is denied.
- logical operator—Select one of these logical operators: EQ (equal to), NE(not equal to), LT (less than GT (greater than), GE (greater then or equal to) LE (less than or equal to).
- right-hand side—Specifies the value that you want CA Top Secret to compare against the contents of the record's left-hand side field. Depending on the left-hand side field data-type (CHAR, PACKED, ZONED, BIN, or HEX) determines how to input the right-hand side data value. Make sure the right-hand side data-type matches the left-hand side field's data-type. If not, access to the requested record can be denied. For data-type of CHAR, enclose the data value in double quotes.

AND/OR

Used to link multiple sel-expressions together. Use the AND when both sel-expressions are true; use OR when either statement is true.

This keyword is used with:

- The commands ADDTO and REPLACE
- The SDT Record exclusively
- MISC3(SDT) authority

Example: SELDATA keyword

This example adds a SELDATA field to the SELECT record called PROBE1 for all the first names of employees that begin with Mike in departments 100 and above:

```
TSS ADDTO(SDT) SELECT(PROBE1)
               SELDATA('IF DEPARTMENT GE "100" AND "MIKE" ')
```

In this example, dept is defined in the SDT RECORD with the data-type of CHAR. Values 0 to 9, A to Z, and special characters are accepted.:

```
SELDATA('IF dept EQ "HR1000"')
```

This example uses * as a wildcard match. If the LHS field dept contains a value that begins with H, and ends with 100, this would result in a match or a true expression:

```
SELDATA('IF dept EQ "H**100"')
```

For data-types of PACKED, ZONED, or BIN, enter *only* decimal digits (0 to 9) without quotes and a maximum of 16 decimal digits. In this example salary is defined in the SDT RECORD with the data-type of PACKED, ZONED, or BIN:

```
SELDATA('IF salary LT 50000')
```

In this example the decimal value is a negative number:

```
SELDATA('IF Total LE -1000')
```

In this example the data-type is HEX:

```
SELDATA('IF Table EQ X"FFFFFFFF"')
```

Parentheses with SELDATA

Use the parentheses in complex expressions to indicate how you want CA Top Secret to evaluate the clause. Parentheses group the left and right-hand sides of an expression and they express which part of a complex expression is the left-hand side and which side is the right-hand side.

Consider the following:

IF CODE = "3A"

This example finds a true condition when CODE equals 3A, or CODE equals 3B and SALARY is less than $25,000:

```
SELDATA=('IF ((CODE = "3A") OR  ((CODE = "3B") AND (SALARY LT 25000)))')
```

This example evaluates as true when CODE equals 3A and SALARY is less than $25,000 or when CODE equals 3B and SALARY is less than $25,000:

```
SELDATA=('IF (((CODE = "3A") OR  (CODE = "3B") AND (SALARY LT 25000)))')
```

You must specify parentheses when you use NOT in an expression.

SELECT Keyword—Record Control

Valid on z/OS, z/VSE, and z/VM.

Use the SELECT keyword to:

- Provide an up to eight-character name by which CA Top Secret references a SELECT record in the SDT Record. SELECT allows the administrator to provide an eight-character label for a (compound) Boolean condition, provided in the associated SELDATA.
- Determine whether SLP permission is invoked, and whether RLP read/update permission is invoked (in the “selread” and “selwrite” SELECT names, respectively).
- Permit or revoke a SELECT record associated with an FCT resource.

Use any of the following methods to control access to SELECT records: Expiration, Facility, Time/Day, and Actions. Specify any or all of the access levels associated with an FCT: SET, INQUIRE, ALL, BROWSE, DELETE, NONE, READ, and UPDATE. If ACCESS is not specified, CA Top Secret defaults to READ access.

When used with the SDT record, this keyword has the following format:

```
TSS ADDTO(SDT) SELECT(sel-name)
               DESCRIPT(descript-name)
               SELDATA('IF [NOT] sel-expression [AND|OR] sel-expression')
```

sel-name

Specifies a user-defined record ID that must be unique for each SELECT record. It can contain letters, numbers, and special characters.

Range: Up to 8 characters

descript-name

Designates an optional user-description field used as a logical name for this record. If the description field contains blanks enclose it in single quotes.

Range: Up to 32 characters

When used with PERMIT commands involving RLP and SLP, this keyword has the following format:

```
TSS PERMIT(acid) FCT(oper)
                 SELECT(selread,selwrite)

TSS PERMIT(acid) {OTRAN(tran)|PPT(program)}
                 SELECT(selread)
```

Capacity of list

One SELECT statement per TSS command

selread

Specifies the SDT SELECT record used as the selection process file accesses of READ and BROWSE.

selwrite

Specifies the SDT SELECT record used as the selection process for file accesses of UPATE(WRITE, REWRITE, DELETE).

Note: It is not necessary to have both a selread and selwrite record for a SELECT statement. If selwrite is omitted, then the SELECT record specified by the selread is given both READ and UPDATE accesses.

selin

Can have an access level of: READ, BROWSE

selout

Can have an access level of: WRITE, UPDATE, DELETE

Note: If only an input record is used on the SELECT statement, it would be permitted UPDATE access.

When used with PERMIT, this keyword has the following format:

```
TSS PERMIT(acid) FCT(oper) SELECT(selin,selout)
```

selin

Specifies the input select record.

selout

Specifies the output select record.

Note: It is not necessary to have both an input and output record for a SELECT statement.

Capacity of list

One SELECT statement per TSS command

This keyword is used with:

- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA (in association with an FCT)
- The commands ADDTO, DELETE, PERMIT, LIST, REMOVE, REPLACE, and REVOKE
- The SDT Record exclusively
- MISC3(SDT) authority

Examples: SELECT keyword

This example creates a SELECT record called PROBE1 in the SDT, selecting departments ranging from 200 through 299:

```
TSS ADDTO(SDT) SELECT(PROBE1)
               SELDATA('IF DEPARTMENT GE "200" AND DEPARTMENT LE "299" ')
```

This example permits a user to access all data from an FCT called PAY and select all records so that departments 1000 through 1999 are chosen:

```
TSS PERMIT(USR01) FCT(PAY)
                  ACCESS(ALL)
                  SELECT('IF DEPT GE "1000" AND DEPT LT "2000" ')
```

This example revokes access:

```
TSS REVOKE(USR01) FCT(PAY)
```

This example permits a user to access all data from an FCT called PAY and select all records so that departments 1000 through 1999 are chosen, but limit user only to update DEPARTMENT 1500:

```
TSS ADDTO(SDT) SELECT(READDEPT)
               SELDATA('IF DEPARTMENT GE "1000" AND DEPARTMENT LT "2000" ')

TSS ADDTO(SDT) SELECT(UPDTDEPT)
               SELDATA('IF DEPARTMENT EQ "1500" ')

TSS PERMIT(USR01) FCT(PAY)
                  ACCESS(ALL)
                  SELECT(READDEPT,UPDTDEPT)
```

This example revokes access:

```
TSS REVOKE(USR01) FCT(PAY)
```

SERIALNUM Keyword—Certificate Serial Number

Valid on z/OS.

Use the SERIALNUM keyword to specify the digital certificate's serial number.

This keyword has the following format:

```
TSS LIST SERIALNUM(serial number)
```

This keyword is used with:

- The commands LIST, REMOVE, and REPLACE
- The USER ACID type
- CERTLIST(MISC4) authority

Example: SERIALNUM keyword

This example identifies the digital certificate by its serial number:

```
TSS LIST(acid) {DIGICERT(8-byte name)} |
               {LABLCERT('label name')} |
               {SERIALNUM(serial number)
               ISSUERDN(issuer's DN)}
```

SESSKEY Keyword—Session Keys

Valid on z/OS.

Use the SESSKEY keyword to:

- Define the encryption key for the application.
- Display the session keys associated with designated LINKIDs in the APPCLU Record or PassTicket applications in the NDT Record.

SESSKEY is a one- to 16-byte hexadecimal “password” unique to each application assigned as a PassTicket. A SESSKEY is required for each PassTicket.

SESSKEY cannot be used with REMOVE. To remove the field from the APPCLU record, use REPLACE and specify keyword().

This keyword has the following format:

```
TSS ADDTO(NDT) PSTKAPPL(application)
               SESSKEY(session_key)

TSS ADDTO(APPCLU) LINKID(id)
                  SESSKEY(session_key)
                  INTERVAL(num)

TSS LIST(NDT) DATA(SESSKEY)
```

This keyword is used with:

- The commands ADDTO, REMOVE, and REPLACE
- The ACID types APPCLU and NDT
- DATA(SESSKEY) authority and either MISC2(APPCLU) or MISC2(NDT) authority depending on the keys designated

Examples: SESSKEY keyword

In this example, the security administrator establishes 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:

```
TSS ADD(APPCLU) LINKID(SYS1.LU01.LU02)
                CONVSEC(CONV)
                SESSKEY(1234)
                INTERVAL(30)
```

In this example, the administrator for the APPCLU Record needs to indicate that the SESSKEY provided for the LU01-LU02 link changes every 14 days:

```
TSS ADD(APPCLU) LINKID(SYS1.LU01.LU02)
                SESSKEY(1234)
                INTERVAL(14)
```

In this example, a PassTicket application for TSO consists of the literal 'TSO' and a four-character SMFID. The SMFID is SYSA; therefore, the PassTicket for that system is TSOSYSA. The session key is 296LFD:

```
TSS ADD(NDT) PSTKAPPL(TSOSYSA)
             SESSKEY(296LFD)
```

This example indicates that the session key for KA180987 is A1B2C3:

```
TSS ADD(NDT) PSTKAPPL(KA180987)
             SESSKEY(A1B2C3)
```

This example lists session keys and associated applications:

```
TSS LIST(NDT) DATA(SESSKEY)
```

SHMEMMAX Keyword—Maximum Shared Memory

Valid on z/OS1.6 and above.

Use the SHMEMMAX keyword to specify the maximum number of bytes of shared memory space that this user can allocate.

This keyword has the following format:

```
TSS ADD(TESTID) SHMEMMAX(value)
```

value

This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT returns a length of 4 and a value of X'FFFFFFFF'. You can remove this field from the record by changing it to a null value (change user SHMEMMAX()).

Range: 0 to 16,777,215 followed letter indicating a multiplier

Maximum petabytes: 16383P

Maximum terabytes: 16776192T

The following multiplier table for SHMEMMAX describes the multiplier value used to calculate the total number of bytes:

Multiplier	Decimal	Binary	Hex
M = Megabyte	1,048,576	2**20	00000000 00100000
G = Gigabyte	1,073,741,824	2**30	00000000 40000000
T = Terabyte	1,099,511,627,776	2**40	00000100 00000000
P = Petabyte	1,125,899,906,842,624	2**50	00040000 00000000

This keyword is used with:

- The commands CREATE, ADD, ADDTO, REMOVE, and REPLACE
- The ACID type USER
- ACID(MAINTAIN) or ACID(CREATE) authority

Examples: SHMEMMAX keyword

This example assigns a user a shared memory limit of 12345 megabytes:

```
TSS ADD(TESTID) SHMEMMAX(12345M)
```

This example removes a shared memory limit from a user:

```
TSS REMOVE(TESTID) SHMEMMAX
```

SIGAUDIT Keyword—SIGVER record program definition

Valid on z/OS.

Use the SIGAUDIT keyword to specify what would trigger the logging of a signature verification event during program load.

This keyword is optional and has the following format:

```
TSS ADD(SIGVER) PROGRAM(PROG01)
                SIGAUDIT(ALL|FAIL|NONE)
```

Valid values

ALL

Specifies that all signature verifications are logged.

FAIL

Specifies that only failed signature verifications are logged.

NONE

Specifies that signature verifications are never logged.

Capacity of list

One keyword per SIGVER program entry.

This keyword is used with:

- The commands ADDTO and REPLACE
- The SIGVER record only
- MISC5(SGVADMIN) authority to maintain the SIGVER record.

Examples: SIGAUDIT keyword

This example adds program PROG01 with FAIL attribute of SIGAUDIT keyword:

```
TSS ADDTO(SIGVER) PROGRAM(prog01) SIGAUDIT(fail)
```

SIGNALG—Certificate Signing Algorithm

Valid on z/OS.

Use the SIGNALG keyword to specify the digital certificate algorithm to be used when generating a new certificate. SIGNALG cannot be specified when NISTECC or BPECC is specified.

The following table shows which signing algorithm will be used based on the SIZE parameter and the key type. SHA-384 and SHA-512 are used for compatibility only with applications that do not support the current signature algorithms.

Note: The selection of a signature algorithm is based on the key size of the signing certificate, not the key size of the certificate being generated.

Signing Algorithm Used	RSA Key Size (in bits)	NISTECC Key Size (in bits)	BPECC Key Size (in bits)
SHA-1	Less than 2048		
SHA-256	2048 or more	192, 224, 256	160, 192, 224, 256, 320
SHA-384		384	384
SHA-512		521	512

This keyword has the following format:

```
TSS GENCERT(acid) SIGNALG(SHA1|SHA256)
```

SHA1

Digital certificate will be generated with SHA1 signing algorithm. This is the default for RSA and DSA certificates with key size 1024 and below.

SHA256

Digital certificate will be generated with SHA256 signing algorithm. This is the default for RSA certificates with key size 2048 or larger.

Note: SHA256 cannot be used when DSA is specified.

The keyword is used with:

- The GENCERT command
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority, MISC4(CERTSITE) for CERTSITE ACID, and MISC4(CERTAUTH) for CERTAUTH ACID.

SIGNMULTI Keyword—Allow Multiple Sign Ons

Valid on z/OS and z/VSE.

Use the SIGNMULTI keyword to allow multiple sign ons for specific ACIDS in a facility with the SIGN(S) sub-option. SIGNMULTI is for use with CICS (TYPE=CICS as the FACILITY option). In order to use the SIGNMULTI feature under CICS, the facility must be defined as SIGN(S) and have SNSCOPE of NONE or CICS in the SIT.

This keyword has the following format:

```
TSS ADDTO(acid) FACILITY(facility)
                SIGNMULTI
```

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC1(USER) authority

Examples: SIGNMULTI keyword

This example allows User01 to signon to any facility multiple times:

```
TSS ADDTO(USER01) FACILITY(ALL)
                  SIGNMULTI
```

This example removes the multiple signon capability:

```
TSS REMOVE(USER01) FACILITY(ALL)
                   SIGNMULTI
```

SIGNWITH Keyword—Specify a Private Key to Sign a New Certificate

Valid on z/OS.

Use the SIGNWITH keyword to specify a digital certificate whose associated private key signs a newly generated certificate. If you do not specify this keyword, CA Top Secret signs the certificate with the private key of the certificate being generated (creating a self-signed certificate).

Note: If no private key is associated with the specified certificate, processing stops.

If DCDSN is specified on the GENCERT command, the SIGNWITH keyword is required.

The following limitations exist:

- Only a CERTAUTH or CERTSITE certificate can be used to sign a CERTAUTH or CERTSITE certificate. A user certificate can be used to sign a certificate only for the same user.
- The length of the SUBJECTN field on the certificate specified for SIGNWITH cannot exceed 229 (if SDNSIZE(255) is specified) or 1007 (if SDNSIZE(1024) is specified).

Self-signed certificates are always trusted, while all other certificates are created with the trust status of the certificate specified with the SIGNWITH keyword. If the specified certificate is not trusted, the product issues an informational message but still generates the certificate.

This keyword has the following format:

```
TSS GENCERT SIGNWITH(acid,digicert)
```

acid

Specifies the ACID who owns the certificate that you are using to sign the newly generated certificate.

digicert

Specifies the digital certificate whose private key signs the newly generated certificate.

The keyword is used with:

- The commands GENCERT, REKEY, and RENEW
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) and MISC4(CERTGEN) authority, MISC4(CERTSITE) for CERTSITE ACID, and MISC4(CERTAUTH) for CERTAUTH ACID

Example: Use SIGNWITH to Specify a Certificate (with Private Key) That Signs a Newly Generated Certificate

This example generates a digital certificate to be signed by the private key of certificate cert0001:

```
TSS GENCERT(user1) DIGICERT(cert0001)
                   DCDSN(user1.cert.data)
                   SIGNWITH(user1,cert0001)
```

SIGREQ Keyword—SIGVER record program definition

Valid on z/OS.

Use the SIGREQ keyword to specify if a signature is required for a SIGVER program.

This keyword is optional and has the following format:

```
TSS ADD(SIGVER) PROGRAM(PROG01) SIGREQ(YES|NO)
```

Capacity of list

One keyword per SIGVER program entry.

This keyword is used with:

- The commands ADDTO and REPLACE
- The SIGVER record only
- MISC5(SGVADMIN) authority to maintain the SIGVER record.

Examples: SIGREQ keyword

This example adds program PROG01 with signature required attribute to the SIGVER Record:

```
TSS ADDTO(SIGVER) PROGRAM(prog01) SIGREQ(YES)
```

This example modifies program PROG01 with signature not required attribute:

```
TSS REP(SIGVER) PROGRAM(prog01) SIGREQ(NO)
```

SITRAN Keyword—CICS Automatic Transaction

Valid on z/OS and z/VSE.

Use the SITRAN keyword to specify which CICS transaction CA Top Secret automatically executes after an ACID successfully signs on to a facility.

Note: If a SITRAN is added to an ACID that already has a CICS transaction defined, the transaction is replaced.

This keyword has the following format:

```
TSS ADDTO(acid) SITRAN(transaction name [,facility])
```

transaction name

Specifies the CICS transaction that automatically executes when an ACID signs on to a facility.

Length: 1 to 8 characters

This keyword is used with:

- The commands CREATE, REPLACE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: SITRAN keyword

This example forces execution of the CICS transaction PAY6 for a profile ACID, regardless of the facility logged on:

```
TSS ADDTO(PROF3C) SITRAN(PAY6)
```

This example causes CA Top Secret to invoke the transaction PAY6 only when logged on to the CICSTEST facility:

```
TSS ADDTO(PROF3C) SITRAN(PAY6,CICSTEST)
```

This example removes the SITRAN value:

```
TSS REMOVE(PROF3C) SITRAN(PAY6,CICSPROD)
```

SMASYS and SMANODE Keywords—Startup, Recovery, and Thread Options

Valid on z/OS.

Use the SMASYS/SMANODE keywords NDT record entries to define:

- The startup options for application login to the CA Top Secret Management Architecture server by CA Top Secret
- The recovery file used for CA Top Secret Management Architecture audit transactions
- The number of threads used by the CA Top Secret Management Architecture interface

This keyword has the following format:

```
TSS ADD(NDT) SMASYS(sysid)
             RECOVERYDSN(dsname)
             RECVDSN(Recovery DSN)
```

Recovery DSN

Specifies the data set used by the CA Top Secret Management Architecture recovery process at SMA initialization. The recovery file must be created, initialized, and catalogued before starting CA Top Secret Management Architecture. Use the the INITSMAR job in the SAMPJCL library to create the Management Architecture recovery file.

The recovery file cannot be shared among systems may not be used for LDS processing.

Size: 44 bytes

ACTIVE(YES|NO)

Indicates the System Management Architecture record is active and communication with the System Management Architecture server specified is attempted.

SMAPSWD(SMA password)

Indicates the application password used with the APPLID and CREDENTL fields for binding to the System Management Architecture server. This field cannot be modeled. This field allows mixed case characters.

Size: 128 bytes

SMAAPPL(Application ID)

The application ID is generated when an application is registered to System Management Architecture.

THRDNUM(number of threads)

The number of threads the address space generates.

Range: 1 to 100

Default: 10

SMACRED(SMA credentials)

The credential generated when an application is registered to System Management Architecture.

Example: SMASYS keyword

This example defines the remote System Management Architecture host for a SMASYS:

```
TSS ADDTO(NDT) SMASYS(sysid)
               SMANODE(nodename)
               ACTIVE(yes|no)
               SMACRED(SMA credentials)
               SMAPSWD(SMA password)
               SMAAPPL(Application ID)
               THRDNUM(number of threads)
```

SMSAPPL Keyword—Maintain Default SMS Application

Valid on z/OS.

Use the SMSAPPL keyword to add or remove a default SMS application identifier.

This keyword has the following format:

```
TSS ADDTO(acid) SMSAPPL(application)
```

application

Specifies a default SMS application identifier.

Capacity: 1 keyword per ACID

Range: 1 to 8 characters

This keyword is used with:

- The commands CREATE, ADDTO, REPLACE, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(SMS) authority

This keyword is used with the ADDTO and WHOHAS commands.

Examples: SMSAPPL keyword

This example gives user JVAVCA a default data application identifier of PAYROLL:

```
TSS ADDTO(JVAVCA) SMSAPPL(PAYROLL)
```

This example removes the SMSAPPL attribute:

```
TSS REMOVE(JVAVCA) SMSAPPL(PAYROLL)
```

Note: If the value for ACSDEFAULTS is YES in the SMS ACS routine, the application information is extracted from CA Top Secret. If you do not use the input variables to ACS routines that are saved in CA Top Secret, set this option to NO. This saves additional calls to CA Top Secret during allocation processing.

SMSDATA Keyword—Default SMS Data Class

Valid on z/OS.

Use the SMSDATA keyword to add or remove a default SMS data class.

This keyword has the following format:

```
TSS ADDTO(acid) SMSDATA(data class name)
```

data class name

Specifies the default SMS data class.

Capacity: One keyword per ACID

Range: 1 to 8 characters

This keyword is used with:

- The commands CREATE, ADDTO, REPLACE, REMOVE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(SMS) authority

This keyword is used with the ADDTO and WHOHAS commands.

Examples: SMSDATA keyword

This example gives user JVAVCA a default data class of ALLDATA:

```
TSS ADDTO(JVAVCA) SMSDATA(ALLDATA)
```

This example removes the SMSDATA attribute:

```
TSS REMOVE(JVAVCA) SMSDATA(ALLDATA)
```

Note: If the value for ACSDEFAULTS is YES, the data set owner information is extracted from CA Top Secret. If you do not use the input variables to ACS routines that are saved in CA Top Secret, set this option to NO. This saves additional calls to CA Top Secret during allocation processing.

SMSMGMT Keyword—Default SMS Management Class

Valid on z/OS.

Use the SMSMGMT keyword to add or remove a default SMS management class.

This keyword has the following format:

```
TSS ADDTO(acid) SMSMGMT(management class name)
```

management class name

Specifies a default SMS management class.

Capacity: 1 keyword per ACID

Range: 1 to 8 characters

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(SMS) authority

This keyword is used with the ADDTO and WHOHAS commands.

Examples: SMSMGMT keyword

This example gives user JVAVCA a default data management class of SYSTCLAS:

```
TSS ADDTO(JVAVCA) SMSMGMT(SYSTCLAS)
```

This example removes the SMSMGMT attribute:

```
TSS REMOVE(JVAVCA) SMSMGMT(SYSTCLAS)
```

Note: If the value for ACSDEFAULTS is YES, the management class information is extracted from CA Top Secret. If you do not use the input variables to ACS routines that are saved in CA Top Secret, set this option to NO. This saves additional calls to CA Top Secret during allocation processing.

SMSSTOR Keyword—Default SMS Storage Class

Valid on z/OS.

Use the SMSSTOR keyword to add or remove a default SMS storage class.

Note: If the value for ACSDEFAULTS is YES, the storage class information is extracted from CA Top Secret. If you do not use the input variables to ACS routines that are saved in CA Top Secret, set this option to NO. This saves additional calls to CA Top Secret during allocation processing.

This keyword has the following format:

```
TSS ADDTO(acid) SMSSTOR(storage class name)
```

storage class name

Specifies the default SMS storage class.

Capacity: 1 keyword per ACID

Range: 1 to 8 characters

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(SMS) authority

This keyword is used with the ADDTO and WHOHAS commands.

Examples: SMSSTOR keyword

This example gives user JVAVCA a default data storage class of SYSSTOR:

```
TSS ADDTO(JVAVCA) SMSSTOR(SYSSTOR)
```

This example removes the SMSSTOR attribute:

```
TSS REMOVE(JVAVCA) SMSSTOR(SYSSTOR)
```

SNAME Keyword—Map User from Lotus Notes

Valid on z/OS and z/VM.

Use the keyword SNAME to map a user identity from Lotus Notes z/OS UNIX to a CA Top Secret ACID.

This keyword has the following format:

```
TSS ADDTO(acid) SNAME('name')
```

name

Specifies Lotus Notes name. Contain within in single quotes.

Length: Up to 64 characters

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, REPLACE and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID (MAINTAIN) authority

Examples: SNAME keyword

This example maps a Lotus Notes z/OS UNIX user identity name to a CA Top Secret ACID:

```
TSS ADDTO(acid) SNAME('Lotus user identity name')
```

This example removes and un-map a Lotus Notes for z/OS UNIX user from an CA Top Secret ACID:

```
TSS REMOVE(acid) SNAME('Lotus user identity name')
```

SOURCE Keyword—Source Prefixes

Valid on z/OS, z/VSE, and z/VM.

Use the SOURCE keyword to specify source reader or terminal prefixes through which the associated ACID may enter the system.

Terminal restriction can be used to restrict Automatic Terminal Signon ACIDs from being used at another terminal.

This keyword has the following format:

```
TSS ADDTO(acid) SOURCE(oper,...)
```

oper

Specifies the source reader or terminal prefixes the ACID may enter the system through.

Length: 1 to 8 eight characters.

Capacity: 1 to 5 prefixes per command

This keyword is used with:

- The commands CREATE, ADDTO and REMOVE
- The START The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: SOURCE keyword

This example forces a user to initiate all jobs from remote 5:

```
TSS ADDTO(R5USR1) SOURCE(R5)
```

This example removes the SOURCE assignment:

```
TSS REMOVE(R5USR1) SOURCE(R5)
```

START Keyword—Activation Date

Valid on z/OS.

Use the START keyword to specify an optional activation date. This data is not the same as the activation date defined in the certificate itself. The web server validates that date. This date gives the security administrator the ability to specify when the certificate will become active on MVS.

This keyword has the following format:

```
TSS ADDTO(acid) DIGICERT(name)
                DCDSN(dsname)
                START(mm/dd/yy)
```

The format for date entries is determined by the settings for the DATE control option. For example DATE(MM DD YY) or DATE(MM,DD,YY) may necessitate specifying the START date within quotes:

```
TSS PERMIT(USER01) DSN(****.FILE)
                   START('05 01 02')
```

This keyword is used with:

- The commands ADDTO and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- No specific authority

Example: START keyword

This example adds a digital certificate to user USER1 that is not active on z/OS until November 15, 2008:

```
TSS ADDTO(USER1) DIGICERT(CERT0001)
                 DCDSN(USER1.CERT.DATA)
                 START(11/15/08)
```

STCACT Keyword—Operator Accountability

Valid on z/OS.

Use the STCACT keyword with the STC special record to invoke operator accountability. When a START command is entered from the console, requests the ACID and password. This suppresses starting the task unless a valid ACID and password have been supplied. Operator accountability generates an audit event.

If the ACID or password is invalid the started task fails. An ACID with password NOPW cannot be used for started task operator accountability.

This keyword has the following format:

```
TSS ADDTO(STC) PROCNAME(stcname|DEFAULT)
               ACID(acid|action)
               [STCACT]
```

stcname

The STC procedure name.

Range: 1 to 8 characters

DEFAULT

Provides undefined started tasks with a default ACID or default action. A started task is undefined if the procname in the operator console START command does not match any of the STC PROCNAME procedures or prefixes on file.

acid

Pre-existing ACID that CA Top Secret assigns the started task with designated PROCNAME. If the ACID is defined with a significant password (other than NOPW), the system console prompts for the PASSWORD associated with "acid." If the ACID is deleted, if the stcname procedure is started and assigned to this now invalid ACID. An error is issued by the delete but the command is allowed to continue.

Range: 1 to 8 characters

action

If no ACID was created for the STC, choose one of the following actions:

- BYPASS—Security checking for the stcname is bypassed. This is not recommended for complex tasks like CICS, where BYPASS security can cause security failure and unpredictable outcomes during initialization and transaction execution.
- FAIL—The stcname initiation fails.
- PROMPT—The console operator is prompted for an ACID and password assigned for this instance of the started task of this stcname.

This keyword is used with:

- The command ADDTO
- The STC ACID only
- MISC9(STC) authority

Examples: STCACT keyword

This example indicates that the use of the disk copy STC is audited:

```
TSS ADDTO(STC) PROCNAME(DISKCOPY)
               ACID(OPS99)
               STCACT
```

This example removes the STCACT attribute by the entire STC definition, and then defines the STC without the attribute:

```
TSS REMOVE(STC) PROCNAME(DISKCOPY)

TSS ADDTO(STC) PROCNAME(DISKCOPY)
               ACID(OPS99)
```

SUBJECTN Keyword—Certificate DN

Valid on z/OS.

Use the SUBJECTN keyword to specify the ACID's distinguished name in a digital certificate. The values override the values contained in the certificate request in the DCDSN data set. The attributes can consist of:

- 229 characters for a self-signed certificate if SDNSIZE(255) control option is specified, 1007 characters if SDNSIZE(1024) control option is specified
- 255 characters for a non self-signed certificate if SDNSIZE(255) control option is specified, 1024 characters if SDNSIZE(1024) control option is specified
- 229 characters for a certificate added to CERTAUTH if SDNSIZE(255) control option is specified, 1007 characters if SDNSIZE(1024) control option is specified

You can use A-Z and 0-9. The exception is C=COUNTRY. This is a 2-digit value field.

If you do not specify DCDSN or SUBJECTN, the SUBJECTN defaults to the ACID's name field.

Notes:

- If any of the values contain blanks, enclose them in double quotes.
- The complete SUBJECTN phrase is enclosed in parentheses and single quotes.
- The attributes are separated by spaces. No matter how many spaces exist between attributes, they count as one space.
- Each attributes has a limit of 64 characters.
- You can specify multiple values for the organizational unit (OU=).

This keyword has the following format:

```
TSS GENCERT SUBJECTN{('CN="common-name"
                      T="title"
                      OU="organizational-unit-name1"
                      O="organizational-name"
                      L="locality"
                      ST="state–or–province"
                      C="2-digit-only-country"')}
```

This keyword is used with:

- The GENCERT and RENEW commands
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN)and MISC4(CERTGEN) authority
- MISC4(CERTSITE) for CERTSITE ACID and MISC4(CERTAUTH) for CERTAUTH ACID

Example: SUBJECTN keyword

This example specifies SUBJECTN:

```
TSS GENCERT(user1) DIGICERT(cert00001)
                   SUBJECTN ('CN="john doe" OU=payroll')
```

This example specifies a SUBJECTN parameter value that contains blanks enclosed within double quotes:

```
TSS GENCERT(user1) DIGICERT(cert00001)
                   SUBJECTN ('CN="john doe" O=Company OU="TSS Development"')
```

This example specifies a SUBJECTN parameter value that contains multiple values for organizational unit:

```
TSS GENCERT(user1) DIGICERT(cert00001)
                   SUBJECTN ('CN="john doe" OU=payroll OU=accounting')
```

SUSPEND Keyword—Prevent Access after Violation

Valid on z/OS, z/VSE and z/VM.

Use the SUSPEND keyword to prevent ACIDs from accessing the system when a violation occurs.

Add the FOR or UNTIL keywords onto a SUSPEND entry to limit the number of days the suspension is enforced. When SUSPEND is used with FOR or UNTIL, the TSS LIST output for the suspended ACID contains the following content:

```
SUSPEND =  UNTIL mm/dd/yy
```

Until an administrator removes the suspension (through the ASUSPEND keyword), the TSS LIST output for the suspended ACID contains the aforementioned text until the ACID completes a successful signon.

Issuing the ADDTO command to suspend a signed-on ACID automatically triggers a type 71 RACF Event Notifications (ENF) signal, which provides notification about the change to the ACID's security record. Applications that receive the signal can take action (for example, refreshing the user's security record in a CICS remote (AOR) region, thereby suspending the ACID immediately).

This keyword has the following format:

```
TSS ADDTO(acid) SUSPEND
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC1(SUSPEND) authority

  Administrators who do not have MISC1(SUSPEND) authority can add a SUSPEND to users within their scope if given UPDATE access to entity TSSCMD.USER.ADDTO.SUSPEND in the CASECAUT resource class

  Administrators who do not have MISC1(SUSPEND) authority can remove a SUSPEND from users within their scope if given UPDATE access to entity TSSCMD.USER.REMOVE.SUSPEND in the CASECAUT resource class

Example: Remove a Temporary Suspension

This example removes a temporary SUSPEND (SUSPEND UNTIL(DATE)):

```
TSS REMOVE(ACARP) ASUSPEND
                  UNTIL
```

Example: Remove the SUSPEND Attribute from an ACID

This example removes the SUSPEND attribute from an ACID (or unsuspends an ACID that was automatically suspended due to a violation):

```
TSS REMOVE(ACARP) SUSPEND
```

The command removes password suspensions, installation exit suspensions, and violation suspensions.

Example: Suspend an ACID for Five Days

This example suspends USER01 for five days.

```
TSS ADDTO(USER01) SUSPEND FOR(5)
```

Example: Suspend an ACID Until a Specific Date

This entry suspends USER01 until December 17, 2017.

```
TSS ADDTO(USER01) SUSPEND UNTIL(12/17/17)
```

Note: If an ACID is suspended administratively, the ASUSPEND keyword must be used to remove the suspension

More information:

ASUSPEND Keyword—Remove Suspension (see page 127)

SYMXCERT Keyword—Symmetric Certificate Labels

Valid on z/OS.

Use the SYMXCERT keyword to specify one or more labels of digital certificate which you may use to export the symmetric keys protected by this profile. Each certificate must exist in the ICSF key store.

This keyword has the following format:

```
TSS PER(acid) CSFKEYS(resource) SYMXCERT(label1,label2,…label12)
```

You can specify up to 12 certificate labels. Each label may be a maximum of 97 characters. An asterisk (*) indicates any certificate in the ICSF key store may be used.

The form of the certificate label is qualifier/label-name.

qualifier

Specifies an optional qualifier in the certificate label string when multiple certificates have the same label. The qualifier is translated to uppercase characters. The meaning of the qualifier depends on where the certificate resides. When the certificate resides in a key ring, the qualifier is the TSS ACID name of the certificate owner. When the certificate resides in a PKCS #11 token, the qualifier is the value of the CKA_ID attribute of the certificate. The CKA_ID value may be up to 64 hexadecimal characters.

/label-name

Specifies the certificate label assigned when the certificate was created. The forward slash (/) must be specified before the certificate label.

This keyword is used with the following:

- The PERMIT command
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- CSFKEYS and XCSFKEY resource classes only
- Resource(XAUTH) authority to specify ACTION for resources that are owned within their scope

Example: SYMXCERT keyword

This example allows you specify one or more certificate labels which may be used to export the symmetric key with the NJDEPT1 ACID:

```
TSS PER(NJDEPT1) CSFKEYS(resource) SYMXCERT(label1)
```

SYMXPORT Keyword—Symmetric Key Export Options

Valid on z/OS.

Use the SYMXPORT keyword to specify which public keys can be used to export a symmetric key that is controlled by the permit. If not specified, any public key is eligible.

This keyword has the following format:

```
TSS PER(acid) CSFKEYS(resource) SYMXPORT(BYANY|BYNONE|BYLIST)
```

BYANY

Any symmetric token may be used subject to existing SAF checks.

BYNONE

The symmetric token may not be exported.

BYLIST

Any symmetric token in either the certificate list or the PKDS label list may be used to export the key.

This keyword is used with:

- The PERMIT command
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- CSFKEYS and XCSFKEY resource classes only
- Resource(XAUTH) authority to specify ACTION for resources that are owned within their scope

Example: SYMXPORT keyword

This example allows you use use the NJDEPT1 ACID with any symmetric token subject to existing SAF checks:

```
TSS PER(NJDEPT1) CSFKEYS(resource) SYMXPORT(BYANY)
```

SYMXKEYS Keyword—Symmetric PKDS Labels

Valid on z/OS.

Use the SYMXKEYS keyword to specify one or more PKDS labels of public keys which you may use to export the symmetric keys protected by this profile. The public keys must reside in the ICSF PKA data set (PKDS).

This keyword has the following format:

```
TSS PER(acid) CSFKEYS(resource) SYMXKEYS(label1,label2,…label12)
```

You may specify up to 12 certificate labels. Each label may be a maximum of 64 characters. An asterisk (*) specifies any public key in the ICSDF PKDS can be used.

This keyword is used with the following:

- The PERMIT command
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- CSFKEYS and XCSFKEY resource classes only
- Resource(XAUTH) authority to specify ACTION for resources that are owned within their scope

Example: SYMXKEYS keyword

This example allows you specify one or more PKDS labels which may be used to export the symmetric key with the NJDEPT1 ACID:

```
TSS PER(NJDEPT1) CSFKEYS(resource) SYMXKEYS(label1)
```

SYSID Keyword—Identify System

Valid on z/OS, z/VSE, and z/VM.

Use the SYSID keyword to identify the system to which the authorization applies. The SYSID is actually the SMFID. You should specify a maximum of four characters for z/OS systems. The keyword supports up to eight characters for VM systems.

This value can be masked only when used on ADD/REMOVE with CRITMAP or LDAPNODE. Do not use SYSID with PROGRAM or IMS resources.

Note: SYSID is valid on an ADDTO/REMOVE command only when used with the FACILITY or CRITMAP keywords.

This keyword has the following format:

```
TSS ADDTO(acid) [FACILITY(facility)]
                SYSID(sysid)
                [CRITMAP(recid)]
```

acid

Specifies the ACID to which the authorization applies.

facility

Specifies the facility to which the ACID should have access.

sysid

Specifies the system on which the authorization applies.

recid

Specifies a unique eight-byte identifier of a CRITMAP record (in the SDT on the security file) that contains defined criteria and acts as a filter to assign an ACID to a user. You use the TSS ADD CRITMAP command to define the criteria.

This keyword is used with:

- The commands ADDTO, PERMIT, REMOVE, REPLACE, and REVOKE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Example: Allow GroupA to Use the TSO Facility on SMFID TS01

This example allows GroupA to use the TSO facility (but only on the system that has an SMFID of TS01):

```
TSS ADDTO(GroupA) FACILITY(TSO)
                  SYSID(TS01)
```

TARGET Keyword—Node Assignment

Valid on z/OS, z/VSE, and z/VM.

Use the TARGET keyword to:

- Specify which CA Top Secret nodes receive commands and how the local node processes it
- Assign nodes associated with ACIDs

TARGET overrides the CPFTARGET control option which sets up default routing instructions for the entire installation. If CPF is active and a command is issued *without* specifying a TARGET, CA Top Secret propagates the command according to the instructions set by CPFTARGET. If CPFTARGET is set to AUTO, the command is routed based on the DEFNODE keyword.

Notes:

- When administering commands against a user-defined resource class, the resource class must be defined on the sending node or the node where the resource class was defined.
- When using the TARGET keyword with a TSS LIST command, specify WAIT(YES) on the TSS LIST command to receive a response from a remote node. If WAIT(YES) is not specified, the TSS LIST command executes locally.

This keyword has the following format:

```
TSS ADDTO(acid) keyword(s)
                TARGET(node,node,...)

TSS LIST(ACID) TARGET(*|LOCAL|nodename)
```

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.

Note: If a command is issued against USER01 and CPFTARGET is set to AUTO and TARGET was not specified on the command, the command is automatically propagated to the NYC, CHI and DET nodes. If CPFTARGET is set to * and TARGET(SELECT) is specified on the command and the CPFNODES control option is CPFNODES(ATL,NYC(NB),CHI(NB),DET(NB)), the command is sent to the NYC, CHI, and DET nodes.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, REPLACE, RENAME, ADMIN, DEADMIN, WHOOWNS, WHOHAS, MOVE, LIST, and DELETE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TARGET) authority

Examples: TARGET keyword

This example adds data sets beginning with SYS1. to the Accounting Department and transmit them to all nodes whose names start with R:

```
TSS ADDTO(ACCDPT) DSNAME(SYS1.)
                  TARGET(R*)
                  WAIT(Y)
```

This example removes the data sets the administrator enters:

```
TSS REMOVE(ACCDPT) DSNAME(SYS1.)
                   TARGET(R*)
                   WAIT(Y)
```

This example grants USER01 ownership of the ABC123 data set, propagates to the local node and all nodes defined in the CPFNODES control option, because a TARGET of * is specified.

```
TSS ADDTO(USER01) DSNAME(ABC123)
                  TARGET(*)
```

This example only executes on the local node (as specified by the =).

```
TSS ADDTO(USER01) DSNAME(ABC123)
                  TARGET(=)
```

The example displays the users on CPU1, CPU2, and the local node that have access to payroll data:

```
TSS WHOHAS DSNAME(PAYROLL.)
           TARGET(CPU1,CPU2)
           WAIT(Y)
```

This example grants ownership of the SYS1 data set prefix to DEPT01 on all remote R-prefixed nodes identified by the CPFNODES control option:

```
TSS ADDTO(DEPT01) DSNAME(SYS1.)
                  TARGET(R*)
                  WAIT(Y)
```

This example displays the users on CPU1, CPU2 and the local node that have access to the PAYROLL data set prefix:

```
TSS WHOHAS DSNAME(PAYROLL.)TARGET(CPU1,CPU2)
           WAIT(Y)
```

THREADS Keyword—Maximum Number of PTHREAD Created Threads

Valid on z/OS.

Use the THREADS keyword to specify the maximum number of pthread_created threads, including those running, queued, and exited but not detached, that a single process can have concurrently active.This field overrides the MAXTHREADS parameter in the BPXPRMxx member of PARMLIB for this user.

THREADS is equivalent to THREADSMAX in RACF

This keyword has the following format:

```
TSS ADD(acidname) THREADS(nnnnnn)
```

nnnnnn

The maximum number of pthread_created threads a single process can have active.

Range: 0 to 100,000

Default: USS takes the system defaults set in BPXPRMxx.

This keyword is used with the ADDTO, REPLACE, REMOVE, and WHOHAS commands.

Examples: THREADS keyword

This example assigns an acid a value of 10 THREADS:

```
TSS ADD(TESTID) THREADS(10)
```

This example removes THREADS from the acid:

```
TSS REMOVE(TESTID) THREADS
```

TIMEREC Keyword—Time Range Label

Valid on z/OS, z/VSE, and z/VM.

Use the TIMEREC keyword to:

- Provide a name to reference a TIMEREC record in the SDT Record. TIMEREC allows the administrator to label one or more time-range specifications.
- To permit or revoke a time restriction on any resource.
- Add, remove, replace, or list TIME records in the SDT Record.

Note: The TIMEREC and TIMES keywords are mutually exclusive on a PERMIT.

This keyword has the following format:

```
TSS ADDTO(SDT) TIMEREC(time-name)
               DESCRIPT(descript-name)
               RANGE(hhmm:hhmm,...)
```

This keyword has the following format for TSS PERMIT/REVOKE:

```
TSS PERMIT(acid) RESCLASS(resource-name)
                 TIMEREC(time-name)
```

time-name

Specifies a time-name that can contain letters, numbers or special characters.

Range: 1 to 8 characters

Capacity: 1 TIMEREC per command

This keyword is used with:

- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- The commands ADDTO, LIST, PERMIT, REMOVE, REPLACE, and REVOKE
- The SDT Record exclusively
- MISC3(SDT) authority

TIMEREC Access

Available access levels are determined by the RDT resource class being permitted. If ACCESS is not specified, CA Top Secret defaults to READ access.

The administrator can use any of the following methods to control access to TIME records: Expiration, Facility, and Actions.

The administrator can specify any or all of the access levels associated with a particular resource. For example, a data set would have the following access levels: ALL, CREATE, CONTROL, FETCH, NONE, READ, SCRATCH, UPDATE, and WRITE.

Example: TIMEREC keyword

This example adds a TIME record called TEMP1 to the SDT:

```
TSS ADDTO(SDT) TIMEREC(TEMP1)
```

This example permits a user to update a data set with a prefix of SFT and using a department's TIME record named TIME1:

```
TSS PERMIT(USR01) DSNAME(SFT.)
                  ACCESS(ALL)
                  TIMEREC(TIME1)
```

This example revokes access:

```
TSS REVOKE(USR01) DSNAME(SFT.)
```

TIMES Keyword—Access Hours

Valid on z/OS, z/VSE, and z/VM.

Use the TIMES keyword to assign a range of hours during which a facility or resource may be accessed.

This keyword has the following format:

```
TSS ADDTO(acid) FACILITY(facility)
                TIMES(nn,nn)

TSS PERMIT(acid) resource(prefix(es))
                 TIMES(nn,nn)
```

This keyword is used with:

- The commands ADDTO, PERMIT, REMOVE, REPLACE, and REVOKE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- No specific authority

Time Ranges

The first two digits in the TIMES operand specify the hour at which (CPU time) CA Top Secret permits access to the resource. The second pair of digits specify the hour through which CA Top Secret permits access. Thus, CA Top Secret permits access from the first minute of the hour specified in the first operand, until the first minute of the hour specified in the second operand.

Note: When the time range wraps around midnight, access is permitted until one minute before the "second" hour expires. For example if the range is (10,20), the user is permitted access from 10 a.m. until 8 p.m. If the range is (20,10), indicating that it wraps around midnight, then the user can access that resource from 8 p.m. until 10:59 a.m. on the following morning.

To specify an inclusive range, set the start time to a lesser number than the stop time. This entry permits USER01 to open the data set between 0600 hours (6:00 a.m.) and 1200 hours (noon).

```
TSS PERMIT(USER01) DSNAME(**.+COPS)TIMES(06,12)
```

To indicate a range that wraps around the 24-hour clock, set the start time greater than the stop time. The entry permits USER01 to access the data set between 8:00 p.m. in the evening and 8:00 a.m. on the following morning.

```
TSS PERMIT(USER01) DSNAME(**.+COPS)TIMES(20,08)
```

To permit access for one hour (10 am to 11 am), enter:

```
TSS PERMIT(acid) DSNAME(**.PREFIX)TIMES(10,11)
```

Examples: TIMES keyword

This example specifies an inclusive range, set the start time to a lesser number than the stop time:

```
TSS ADDTO(USER01) FACILITY(CICS)
                  TIMES(06,12)
```

This example allows USER01 to access the CICS facility from 8 p.m. until 10 a.m. the following day:

```
TSS ADDTO(USER01) FACILITY(CICS)
                  TIMES(20,10)
```

This example assigns access for one hour (10 a.m. to 11 a.m.):

```
TSS ADDTO(acid) FACILITY(CICS)
                TIMES(10,11)
```

TOKENADD Keyword—Create PKCS#11 Token

Valid on z/OS.

Use the TOKENADD keyword with the P11TOKEN function to create a PKCS#11 token.

This keyword has the following format:

```
TSS P11TOKEN TOKENADD LABLCTKN(token name)
```

LABLCTKN(token name)

Specifies the name of the token being created. The token must not already exist. The first character must be alphabetic or national. Permitted characters are: alphanumeric, national and a period. Lower case letters are folded to upper case.

Range: Up to 32 characters

This keyword is used with MISC3(PTOK) authority.

Also controlled by ICSF using resources in the CRYPTOZ class. Authority to resources in the IBMFAC class is not required.

Example: TOKENADD keyword

This example creates a token named TOKEN#1:

```
TSS P11TOKEN TOKENADD LABLCTKN(token#1)
```

TOKENDEL Keyword—Delete PKCS#11 Token

Valid on z/OS.

Use the TOKENDEL keyword with the P11TOKEN function to delete a PKCS #11 token.

The command has the following format:

```
TSS P11TOKEN TOKENDEL
             LABLCTKN(token name)
             [FORCE]
```

LABLCTKN(token name)

Specifies the name of the token being deleted. The token must already exist.

FORCE

(Optional) Deletes the token even if an object (certificate, public key, or private key) in the token is not defined to CA Top Secret. This allows an administrator to delete a certificate not defined to CA Top Secret.

This keyword is used with MISC3(PTOK) authority.

Also controlled by ICSF using resources in the CRYPTOZ class. Authority to resources in the IBMFAC class is not required.

Example: TOKENDEL keyword

This example deletes a token called token#1:

```
TSS P11TOKEN TOKENDEL LABLCTKN(token#1)
```

TOKENLST Keyword—PKCS#11 Token Certificate Object Information

Valid on z/OS.

Use this keyword with the P11TOKEN function to display information about certificate objects in a specific PKCS#11 token.

This keyword has the following format:

```
TSS P11TOKEN TOKENLST LABLCTKN(token name)
```

LABLCTKN(token name)

The name of the token being listed. Use the * character at the end of the name to indicate a mask or by itself to list all tokens.

This keyword is used with MISC3(PTOK) authority.

Also controlled by ICSF using resources in the CRYPTOZ class and by authorization to resource IRR.DIGTCERT.LIST in the IBMFAC class.

To list RACF information using this keyword the authority required:

- For one's own certificate is sufficient authority to CRYPTOZ resources and READ authority to IRR.DIGTCERT.LIST
- For some else's certificate is sufficient authority to CRYPTOZ resources and UPDATE authority to IRR.DIGTCERT.LIST
- For CERTSITE or CERTAUTH certificates is sufficient authority to CRYPTOZ resources and CONTROL authority to IRR.DIGTCERT.LIST

Example: TOKENLST keyword

This example displays information about the token LV2TOKEN:

```
TSS P11TOKEN TOKENLST LABLCTKN(LV2TOKEN)
```

TRACE Keyword—Diagnostic Trace Activation

Valid on z/OS, z/VSE, and z/VM.

Use the TRACE keyword to activate a diagnostic trace on all ACID activity (initiations, resource access, violations, user's security mode).

Where CA Top Secret records trace information depends on the settings of the SECTRACE control option.

This keyword has the following format:

```
TSS ADDTO(acid) TRACE
```

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC9(TRACE) authority

Examples: TRACE keyword

This example activates a diagnostic trace to solve an authorization problem:

1. Enter the command:
   ```
   TSS ADDTO(USER01) TRACE
   ```
2. Enter the command:
   ```
   TSS MODIFY(SECTRACE(ACT,WTO|WTL))
   ```

   ACT

   Activates the trace.

   WTO

   Sends trace information to the security console.

   WTL

   Sends trace information to SYSLOG.

   When USER01 runs a job, trace information is sent to the destination(s) specified in Step 2.

For information on how to interpret the resulting trace, see the *Troubleshooting Guide* .

This example removes the TRACE attribute:

```
TSS REMOVE(USER01) TRACE
```

TRANSACTIONS Keyword—Specific Transaction

Valid on z/OS and z/ VSE.

Use the TRANSACTIONS keyword to confine ACIDs to using a specific transaction, or subset of the transactions, available within that facility.

For purposes of LCF protection, facility entities USER77 through USER221 are not considered unique.

The TRANSACTIONS keyword is inclusive in that it confines an ACID to using specific transactions for a facility. See the XTRANSACTIONS keyword to determine how to allow an ACID to use all but a specific list of transactions for a facility.

This keyword has the following format:

```
TSS ADDTO(acid) TRANSACTIONS(facility,(transactions[(G)])
```

facility

Specifies the facility.

transactions

Specifies the transaction, or subset of the transactions, available within a facility.

Size: 1 to 8 characters

Capacity: 1 to 30 transaction names or prefixes per command.

This keyword is used with:

- The commands CREATE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ALL Record
- MISC1(LCF) authority to ADDTO or REMOVE TRANSACTIONS from ACIDs. MISC9(GLOBAL) to ADDTO or REMOVE the TRANSACTIONS resource class to or from the ALL record.

Examples: TRANSACTIONS keyword

This example confines a group of CICS clerks to a set of transactions:

```
TSS ADDTO(PROF1) TRANSACTIONS(CICSP,(CPO1,CPO2,CPO3))
```

The letter G next to the transaction prefix indicates that the ACID is confined to using transactions beginning with the prefix specified. The following command indicates that ACID EJ001 is confined to using transactions prefixed by CPO.

```
TSS ADDTO(EJ001) TRANSACTIONS(CICSP,(CPO(G)))
```

This example removes the TRANSACTIONS restriction:

```
TSS REMOVE(EJ001) TRANSACTIONS(CICSP,(CPO(G)))
```

For purposes of LCF protection, facility entities USER77 through USER221 are not considered unique. For example, if userxxx in the following command is chosen from USER77 through USER221, then the command will affect all facilities from USER77 through USER221.

```
TSS ADDTO(USER1)TRANSACTIONS(user xxx,(TRAN1))
```

If useryyy is a different facility chosen from USER77 through USER221, then both facilities userxxx and useryyy is affected by the LCF definition shown above or the definition shown next.

```
TSS ADDTO(USER1) TRANSACTIONS(useryyy,(TRAN2))
```

TRUST Keyword—Associate a Certificate to a User

Valid on z/OS.

Use the TRUST keyword to associate a digital certificate with a user.

Note: HITRUST is only valid for the ACID named CERTAUTH.

This keyword has the following format:

```
TSS ADDTO(acid) DIGICERT(name)
                TRUST|NOTRUST|HITRUST
```

TRUST

Specifies that the certificate is valid for the user, site, or CA and the private key is not compromised.

On a:

- USER certificate—TRUST indicates that the certificate can be used to authenticate an ACID.
- CERTSITE certificate—TRUST indicates that the certificate can be used without authenticating it.
- CERTAUTH certificate—TRUST indicates that the certificate can be used to authenticate other certificates.

NOTRUST

Indicates that the certificate is not trusted.

HITRUST

Specifies that the certificate is both highly trusted and trusted. Certificate usage applying to trusted certificates also applies to highly trusted certificates. Only CA certificates (CERTAUTH) can be highly trusted.

The trust status is set to the CA's trust status if the:

- Certificate’s signature can be verified.
- Certificate has not expired.
- Certificate’s validity dates fall within the range of the signing certificate.

The default trust status for self-signed certificates is TRUST.

The trust status is set to NOTRUST if the certificate being added or generated:

- Has expired
- Has an invalid date range
- Cannot be verified

The trust status of the new certificate is set to TRUST if the trust status comes from a signing certificate with HITRUST.

If TRUST is not specified and the certificate is signed by another certificate:

- If all of the following conditions are met, the status is set to TRUST:
  - The signing certificate can be located in the database.
  - The signing certificate is a Certification Authority (CERTAUTH).
  - The signing certificate has not expired.
  - If the signing certificate's signature is valid.
  - The validity dates of the certificate being added fall within the range of the signing certificate's validity dates.
- If any of the above conditions are not met, the certificate is inserted as NOTRUST and a message is issued stating why.

This keyword is used with:

- The commands ADDTO and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority

Example: TRUST keyword

This example associates a digital certificate with the name CERT0001 to USER1:

```
TSS ADDTO(USER1) DIGICERT(CERT0001)
                 DCDSN(cert.dataset.p12
                 TRUST
```

This example removes the digital certificate association:

```
TSS REPLACE(USER1) DIGICERT(CERT0001)
                   NOTRUST
```

TSOCOMMAND Keyword—Default TSO Logon Command

Valid on z/OS.

Use the TSOCOMMAND keyword to provide a default command issued at TSO logon.

The addition of a default command will not automatically give an ACID use of a command protected by the PROGRAM. If the default command is protected by PROGRAM or LCF, the ACID must also be permitted to use PROGRAM.

All TSO UADS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) TSOCOMMAND(nnn)
```

nnn

Specifies the default command issued at TSO logon.

Length: 1 to 80 characters

Capacity: 1 command per ACID

If a single quote is part of the name and the entire character string is enclosed in single quotes, use two single quotes together to represent each single quote with the string. For example, to issue the command EX 'FPRST.ADV.LOGON' enter:

```
TSS ADDTO(acid) TSOCOMMAND('EX ''FPRST.ADV.LOGON'' ')
```

This keyword is used with:

- The commands REPLACE, ADDTO, and REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TSO) authority

Examples: TSOCOMMAND keyword

This example adds a command of SDSF to ACID USERA:

```
TSS ADDTO(USERA) TSOCOMMAND(SDSF)
```

This example removes a default logon command of SDSF from USERA:

```
TSS REMOVE(USERA) TSOCOMMAND(SDSF)
```

TSODEST Keyword—TSO Default Destination Identifier

Valid on z/OS.

Use the TSODEST keyword to provide a default destination identifier for TSO generated JCL for TSO users.

All TSO UADS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) TSODEST(id)
```

id

Specifies the default destination identifier for TSO generated JCL.

Length: 1 to 8 characters

Capacity: 1 identifier per ACID

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TSO) authority

Examples: TSODEST keyword

This example adds a destination ID of SYSTEME to ACID USERA:

```
TSS ADDTO(USERA)TSODEST(SYSTEME)
```

This example removes a default destination of SYSTEME from USERA:

```
TSS REMOVE(USERA)TSODEST(SYSTEME)
```

TSOHCLASS Keyword—TSO Default Hold Class

Valid on z/OS.

Use the TSOHCLASS keyword to assign a default hold class for TSO-generated JCL for TSO users.

All TSO UADS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) TSOHCLASS(class)
```

class

Specifies the default hold class for TSO generated JCL.

Length: 1 character

Capacity: 1 class per ACID

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TSO) authority

Examples: TSOHCLASS keyword

This example adds a class of X to ACID USERA:

```
TSS ADDTO(USERA) TSOHCLASS(X)
```

This example removes a default class of X from USERA:

```
TSS REMOVE(USERA) TSOHCLASS(X)
```

TSOJCLASS Keyword—TSO Default Job Class

Valid on z/OS.

Use the TSOJCLASS keyword to assign a default job class for TSO generated job cards from TSO users.

All TSO UADS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) TSOJCLASS(class)
```

class

Specifies the default job class for TSO generated job cards.

Length: 1 character

Capacity: 1 class per ACID

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TSO) authority

Examples: TSOJCLASS keyword

This example adds a class of A to ACID USERA:

```
TSS ADDTO(USERA)TSOJCLASS(A)
```

This example removes a default class of A from USERA:

```
TSS REMOVE(USERA)TSOJCLASS(A)
```

TSOLPROC Keyword—TSO Default Proc

Valid on z/OS.

Use the TSOLPROC keyword to provide a default proc used for TSO logon.

The addition of a default proc will not automatically give an ACID use of a proc protected by TSOPROC. If the default proc is protected, the ACID must also be permitted to use the TSOPROC.

All TSO UADS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) TSOLPROC(proc)
```

proc

Specifies the default proc used for TSO logon.

Prefix length: 1 to 8 characters

Capacity: 1 proc per ACID

The keyword is used with:

- This commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TSO) authority

Examples: TSOLPROC keyword

This example adds a proc of IJKPROC to ACID USERA:

```
TSS ADDTO(USERA) TSOLPROC(IJKPROC)
```

This example removes a default proc of IJKPROC from ACID USERA:

```
TSS REMOVE(USERA) TSOLPROC(IJKPROC)
```

TSOLSIZE Keyword—TSO Default Region Size

Valid on z/OS.

Use the TSOLSIZE keyword to assign a default region size (in kilobytes) for TSO. The addition of a default region size will not automatically give an ACID use of a region size protected by TSOMSIZE. If the default region size is protected by TSOMSIZE, the ACID must also be permitted to use TSOMSIZE.

All TSO UADS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) TSOLSIZE(nnnnnnn)
```

nnnnnnn

Specifies the a default region size for TSO.

Prefix length: 1 to 7 digits

Capacity: 1 region size per ACID

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TSO) authority

Examples: TSOLSIZE keyword

This example adds a region size of 1024K to USERA:

```
TSS ADDTO(USERA)TSOLSIZE(1024)
```

This example removes the default region size for USERA:

```
TSS REMOVE(USERA)TSOLSIZE()
```

TSOMCLASS Keyword—TSO Default Message Class

Valid on z/OS.

Use the TSOMCLASS keyword to assign a default message class for TSO generated JCL for TSO users. All TSO UADS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) TSOMCLASS(class)
```

class

Specifies the default message class for TSO generated JCL.

Prefix length: 1 character

Capacity: 1 class per ACID

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TSO) authority

Examples: TSOMCLASS keyword

This example adds a class of X to ACID USERA:

```
TSS ADDTO(USERA) TSOMCLASS(X)
```

This example removes a default message class of X from USERA:

```
TSS REMOVE(USERA) TSOMCLASS(X)
```

TSOMPW Keyword—Support Multiple TSO UADS Passwords

Valid on z/OS.

Use the TSOMPW keyword to support multiple TSO UADS passwords, on a user-by-user basis.

With TSOMPW, the ACID must first enter his CA Top Secret password, and then he is prompted to enter his UADS password.

All TSO UADS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) TSOMPW
```

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and WHOHAS
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: TSOMPW keyword

This example indicates that CA Top Secret will support multiple passwords for USER01:

```
TSS ADDTO(USER01)TSOMPW
```

This example removes the TSOMPW attribute:

```
TSS REMOVE(USER01)TSOMPW
```

Note: This attribute is ignored if CA Top Secret is maintaining UADS definitions for the User ACID.

TSOMSIZE Keyword—TSO Maximum Region Size

Valid on z/OS.

Use the TSOMSIZE keyword to define the maximum region size (in kilobytes) that a TSO user may specify at logon. All TSO UADS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) TSOMSIZE(nnnnnnn)
```

nnnnnnn

Specifies the maximum region size that a TSO user can specify at logon.

Length: 1 to 7 digits

Capacity: 1 region size per ACID

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TSO) authority

Examples: TSOMSIZE keyword

This example adds a maximum region size of 4096K to USERA:

```
TSS ADDTO(USERA)TSOMSIZE(4096)
```

This example deletes a maximum region size for USERA:

```
TSS REMOVE(USERA)TSOMSIZE()
```

TSOOPT Keyword—Assign Default TSO Options

Valid on z/OS.

Use the TSOOPT keyword to assign default options that a TSO user may specify at logon. All TSO UADS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword has the following format:

```
TSS ADDTO(acid) TSOOPT(option,...)
```

option

Specifies the default options a TSO user can specify at logon.

Prefix length: 1 to 9 nine characters

Capacity of list: 5 options per TSO command

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, REPLACE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TSO) authority

Examples: TSOOPT keyword

This example adds options of NOMAIL and NONOTICES to USERA:

```
TSS ADDTO(USERA) TSOOPT(NOMAIL,NONOTICES)
```

This example deletes an option of NOMAIL for USERA:

```
TSS REMOVE(USERA) TSOOPT(NOMAIL)
```

The following values for TSOOPT are honored by TSO:

- MAIL/NOMAIL
- NOTICES/NONOTICES
- OIDCARD/NOOIDCARD

TSOSCLASS Keyword—TSO Default SYSOUT Class

Valid on z/OS.

Use the TSOSCLASS keyword to assign a default SYSOUT class for TSO generated JCL for TSO users.

This keyword has the following format:

```
TSS ADDTO(acid) TSOSCLASS(class)
```

class

Specifies the default SYSOUT class for TSO generated JCL.

Length: 1 character

Capacity: 1 class per ACID

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TSO) authority

Examples: TSOSCLASS keyword

This example adds a class of A to USERA:

```
TSS ADDTO(USERA)TSOSCLASS(A)
```

This example deletes a class of A from USERA:

```
TSS REMOVE(USERA)TSOSCLASS(A)
```

TSOUDATA Keyword—TSO Data Field

Valid on z/OS.

Use the TSOUDATA keyword to assign a site-defined data field to a TSO user.

This keyword has the following format:

```
TSS ADDTO(acid) TSOUDATA(data field)
```

data field

Specifies the data field for a TSO user.

Prefix length: Four hexadecimal (0 to 9, A to F) characters. If less than four digits are used the field is padded to the right with hex zeros.

Capacity of list: One field per ACID

Note: All TSO UADS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TSO) authority

Examples: TSOUDATA keyword

This example adds a data field of 123A to USERA:

```
TSS ADDTO(USERA) TSOUDATA(123A)
```

This example adds a data field of AB00 to USERA:

```
TSS ADDTO(USERA) TSOUDATA(AB00)
```

This example deletes a data field of 12A from USERA:

```
TSS REMOVE(USERA) TSOUDATA(12A)
```

TSOUNIT Keyword—TSO Default Unit Name

Valid on z/OS.

Use the TSOUNIT keyword to assign a default unit name for dynamic allocations under TSO.

This keyword has the following format:

```
TSS ADDTO(acid) TSOUNIT(name)
```

name

Specifies the default unit name for dynamic TSO allocations.

Prefix length: 1 to 8 characters

Capacity: 1 name per ACID

Note: All TSO UADS resources must be added to a User Record and cannot be added to a Profile or to the ALL Record.

This keyword is used with:

- The commands ADDTO, CREATE, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC2(TSO)

Examples: TSOUNIT keyword

This example adds a unit name of SYSDA to ACID USERA:

```
TSS ADDTO(USERA)TSOUNIT(SYSDA)
```

This example delete a unit name of SYSDA from USERA:

```
TSS REMOVE(USERA)TSOUNIT(SYSDA)
```

TYPE Keyword—ACID Type

Valid on z/OS, z/VSE, and z/VM.

Use the TYPE keyword to specify the type of ACID.

When used with CREATE, this keyword has the following format:

```
TSS CREATE(acid) TYPE(USER|PROFILE|GROUP|DEPARTMENT|DIVISION|ZONE|DCA|
                      VCA|ZCA|LSCA|SCA)
                 NAME('name')
```

If the administrator fails to enter the TYPE keyword, CA Top Secret automatically defaults to TYPE(USER).

When used with LIST, this keyword has the following format:

```
TSS LIST(ACIDS) DATA(dataType(s))
                TYPE(USER|PROFILE|GROUP|DCA|VCA|SCA|ZCA|LSCA|
                     DEPARTMENT|DIVISION|ZONE)
```

CA Top Secret only displays data concerning ACIDs within the administrator's scope. For example, an MSCA or an authorized SCA can list data for the entire site, a VCA for his division and all subordinate departments, and a DCA for his department.

If no operand for the TYPE keyword is entered, CA Top Secret lists all acids within the administrator's scope.

This keyword is used with:

- The commands CREATE, LIST, and MOVE
- The ACID types ACIDS, User, Profile, Group, Department, Division, Zone, VCA, ZCA, LSCA, and SCA

Example: TYPE keyword

This example creates a new department in the PARTS Division:

```
TSS CREATE(PART10) NAME('PARTS RETURN')
                   TYPE(DEPARTMENT)
                   DIVISION(PARTSDIV)
```

This example requests all security information for DCA's in the TECH Division:

```
TSS LIST(ACIDS) DATA(ALL)
                DIVISION(TECH)
                TYPE(DCA)
```

TZONE Keyword—Physical Time Zone

Valid on z/OS and z/VM.

Use the TZONE keyword to specify an ACID's physical time zone in relation to the CPU's time zone. This forces all date and time rules to be based on the ACID's local date and time, not on the CPU's date and time.

This keyword has the following format:

```
TSS ADDTO(acid) TZONE([-]nn)
```

nn

From -12 to +12.

This keyword is used with:

- The commands CREATE, REPLACE, ADDTO, and REMOVE
- ACID(MAINTAIN) authority

Examples: TZONE keyword

In this example, the CPU resides in New York and the user ZAP99 resides in San Diego. To ensure that all date and time checking is based on ZAP99's Pacific Coast Time zone and not the CPU's Eastern Standard Time, enter:

```
TSS ADDTO(ZAP99)TZONE(-3)
```

This example lets the user's timezone revert to the CPU's timezone:

```
TSS REMOVE(ZAP99)TZONE(0)
```

UID Keyword—Assign a User ID to a USS User

Valid on z/OS and z/VM.

Use the UID keyword to assign a numeric user ID (UID) value to each user for security within USS. You can assign UID(0) (to gives the user superuser privileges) or you can assign a unique positive number to the ACID. You can explicitly specify a range for the assignment, or you can accept the range default in the DFLTRNGU control option.

Important! After you add a UID to an ACID, specify TSS MODIFY(OMVSTABS) to perform a refresh and allow the assignment to take effect. Attempting to use the UID assignment before an OMVSTABS refresh could interfere with the ACID's attempts to use USS services.

This keyword has the following format:

```
TSS ADDTO(acid) UID(USS_user_id)

TSS ADDTO(acid) UID(?)
              [RANGE(low_uid,high_uid)]
```

acid

Specifies the ACID to which you are adding a USS UID.

USS_user_id

Specifies the numeric user ID to assign to the ACID. UID 0 has special meaning ("superuser") and can be assigned to multiple ACIDs.

Important! You can assign a specific UID to only one specific ACID. For example, only USERX can be assigned UID 8.

Range: 0 to 2147483647

?

Instructs CA Top Secret to automatically assign the USS UID. If you specify RANGE in the command, the product assigns the first unused positive integer that is greater than or equal to the *low_uid* value and less than or equal to the *high_uid* value. If you do not specify RANGE, the product assigns the first available integer in the range defined by DFLTRNGU control option.

Note: You *cannot* use the ? value with REMOVE.

If you are using the Command Propagation Facility (CPF) and CPFAUTOUID(YES), the TSS command is transmitted with the assigned UID value instead of the ? value (for example, TSS ADDTO(JONATHAN) UID(256) instead of TSS ADDTO(JONATHAN) UID(?)).

Note: For complete information about the CPFAUTOUID control option, see the *CA Top Secret Control Options Guide*.

RANGE

Specifies a range from which the product will choose an available UID (if you have also specified ? in the command syntax to automatically assign a UID).

low_uid

Indicates the lowest assignable UID available to this ACID.

Range: 1 to 2147483647

high_uid

Indicates the highest assignable UID available to this ACID.

Range: 1 to 2147483647

This keyword is used with:

- The commands ADDTO, REMOVE, REPLACE, WHOHAS, and WHOOWNS
- The ACID types SCA, LSCA, ZCA, VCA, DCA, and User
- ACID(MAINTAIN) authority

Example: Assign a Specific UID to an ACID

This example indicates that user ACID USER02 has a UID of 83234:

```
TSS ADDTO(USER02) UID(83234)
```

Example: Remove a UID Assignment from an ACID

This example removes the connection between a UID and an ACID:

```
TSS REMOVE(USER02) UID(83234)
```

Example: Automatically Assign a UID Through the DFLTRNGU Control Option

This example automatically assigns a number to an ACID (using an available integer in the default range defined by the DFLTRNGU control option):

```
TSS ADDTO(johndoe) UID(?)
```

Example: Automatically Assign a UID By Using an Available Integer in a Range That You Specify

This example automatically assigns a number to an ACID within the inclusive range 10,000–200,000:

```
TSS ADDTO(johndoe) UID(?)
                   RANGE(10000,200000)
```

UNAME Keyword—Map User from Novell Directory Services

Valid on z/OS and z/VM.

Use the UNAME keyword to map a user identity from Novell Directory Services to an ACID.

This keyword has the following format:

```
TSS ADDTO(acid) UNAME('uname')
```

UNAME

Specifies the Novell Directory Services user identity name in single quotes.

Length: Up to 246 characters including spaces.

This keyword is used with:

- The commands ADDTO, REMOVE, REPLACE, and WHOHAS
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID (MAINTAIN) authority

Examples: UNAME keyword

This example maps a Novell Directory Services z/OS user identity name to an ACID:

```
TSS ADDTO(acid) UNAME('Novell Dir. Services User identityname')
```

This example removes and un-maps a Novell Directory Services z/OS user from an ACID:

```
TSS REMOVE(acid) UNAME('Novell Dir. Services User identityname')
```

UNBIND Keyword—Remove Certificate from PKCS#11 Token

Valid on z/OS.

Use the UNBIND keyword with the P11TOKEN function to remove a certificate from a PKCS#11 token.

The certificate to be removed is identified by one of:

- The certificate label name and userid (certificate must be defined to CA Top Secret)
- The sequence number in the token (certificate does not have to be defined to CA Top Secret)

The keyword has the following format:

```
TSS P11TOKEN UNBIND
             LABLCTKN(token name)
             SEQNUM(nnnnnnnn)|LABLCERT(certificate label)
             TOKNUSER(userid)
             FORCE
```

LABLCTKN(token name)

Specifies the name of the token being deleted. The token must already exist.

SEQNUM(nnnnnnnn)

Specifies the sequence number of the certificate to remove from the token. If the certificate is not defined to CA Top Secret, then FORCE must be specified. LABLCERT and SEQNUM are mutually exclusive.

LABLCERT(certificate label)

Specifies the digital certificate label name of the certificate to unbind. TOKNUSER must also be specified. LABLCERT and SEQNUM are mutually exclusive.

TOKNUSER(userid)

Specifies the userid for the digital certificate.

FORCE

Specifies that the certificate is to be removed even if it is not defined to CA Top Secret. May only be used when SEQNUM is specified.

This keyword is used with MISC3(PTOK) authority.

Controlled by ICSF using resources in the CRYPTOZ class. Authority to resources in the IBMFAC class is not required.

UNDERCUT Keyword—Transfer Resource Ownership

Valid on z/OS, z/VSE, and z/VM.

Use the UNDERCUT keyword to transfer resource ownership from one ACID to another.

Undercutting occurs when an administrator ADDS a prefix which is shorter than another resource prefix which is already owned. The use of the UNDERCUT keyword indicates that the transfer is intentional. If the UNDERCUT keyword is omitted, CA Top Secret issues an error message.

This prevents CA Top Secret administrators from unintentionally transferring or undercutting ownership.

When resource ownership is transferred from one owner to another, CA Top Secret automatically permits the old owner to have full access to the resource. To prevent the old owner from having full access to resources they no longer own use NOPERMIT.

A data set defined as SYS.01 is owned by ACID2. Since CA Top Secret supports generic prefixing, it will consider SYS.01 and SYS.01.02 to be generic prefixes for the same data set.

An administrator can only undercut prefixes within his scope of authority.

Undercutting only applies to prefixes, not to full resource names. Therefore, if USER. and USER01. were full minidisk names they could exist as separate minidisks.

Undercutting only applies to prefixes of the same resource type. Thus USER could exist as a minidisk, and USER01 could exist as a program.

This keyword has the following format:

```
TSS ADDTO(acid) resource(prefix)
                UNDERCUT
```

This keyword is used with:

- The commands CREATE and ADDTO
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- RESOURCE(OWN) authority

Examples: UNDERCUT keyword

This example transfers ownership of SYS.01.02 from ACID2 to ACID1.

```
TSS ADDTO(ACID1) DSNAME(SYS.01)
                 UNDERCUT
```

This example prevents the old owner from having full access to resources they no longer own:

```
TSS ADDTO(NEWOWNER) DSNAME(SYS.01)
                    UNDERCUT
                    NOPERMIT
```

UNTIL Keyword—Expire Date

Valid on z/OS, z/VSE, and z/VM.

Use the UNTIL keyword to assign or remove the specific date:

- On which an ACID expires.
- When permission to access a resource expires.

The format for date entries is determined by the settings for the DATE control option. The default is mm/dd/yy.

Any year (yy) entered as 70 or above is considered a 20th century date. Any year below 70 is considered a 21st century date. For example, 68 would be processed as being 2068 not 1968.

If the expiration date for the profile has already passed, UNTIL is displayed as “EXPIRED”.

This keyword has the following format:

```
TSS ADDTO(acid) UNTIL(mm/dd/yy)
```

```
TSS PERMIT(acid) resource(prefix(es))
                 UNTIL(mm/dd/yy)
```

This keyword is used with:

- The commands ADDTO, CREATE, PERMIT, REMOVE, REPLACE, and REVOKE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- No specific authority

Examples: UNTIL keyword

The following examples assume the date format is mm/dd/yy.

This example allows use of an ACID until December of 1997:

```
TSS ADDTO(YURGO) UNTIL(12/01/97)
```

This example changes YURGO's expiration date from 12/01/97 to 07/04/98.

```
TSS ADDTO(YURGO) UNTIL(07/04/98)
```

The same change could have been made with the REPLACE function. This example removes the expiration dates:

```
TSS REMOVE(YURGO) UNTIL
```

This example removes the expiration interval when you are unsure of whether you specified FOR or UNTIL on the TSS ADDTO:

```
TSS REMOVE(acid) EXPIRE
```

This example reactivates an expired ACID:

```
TSS ADDTO(ACID56) UNTIL(mm/dd/yy)
```

This example reactivates an expired ACID:

```
TSS ADDTO(USER56) FOR(0)
```

This example uses the UNTIL keyword in conjunction with the FACILITY and PROFILE keywords:

```
TSS ADDTO(USER56) FACILITY(CICSPROD)
                  UNTIL(07/04/97)
```

This example specifying the UNTIL keyword with PROFILE, lets you see the expiration date(s):

```
TSS LIST(profacid) DATA(EXP)
```

The UNTIL keyword cannot be used to REMOVE an expiration date from a facility or profile.

This example will not remove the expiration date from the profile, but will remove the profile from the user record:

```
TSS REMOVE(USER56) PROFILE(TECHPROF)
                   UNTIL
```

To remove an expiration date from a facility or profile, the facility or profile must be removed from the ACID and then added again to the ACID without the expiration date.

This example permits USER01 to READ (default) to any data set suffixed by .FILE until May 1, 2001 (11:59:59 pm on April 30, 2001):

```
TSS PERMIT(USER01) DSNAME(****.FILE)
                   UNTIL(05/01/01)
```

USAGE Keyword—Certificate Trust Level

Valid on z/OS.

(Optional) Use the USAGE keyword to indicate the trust level for a digital certificate being added to a key ring.

This keyword has the following format:

```
TSS ADDTO(acid) KEYRING(8-byte name)
                [LABLRING(237-byte ring name)]
                {RINGDATA(acid,digicert)}
                {RINGDATA(CERTSITE,digicert)}
                {RINGDATA(CERTAUTH,digicert)}
                [DEFAULT]
                [USAGE(PERSONAL|CERTSITE|CERTAUTH)]
```

PERSONAL

Specifies that the certificate is generated for a user.

CERTSITE

Specifies a higher level of trust, for example, from the installation site.

CERTAUTH

Specifies the highest level of trust, for example, from a certificate authority.

This keyword is used with:

- The commands ADDTO, P11TOKEN, REMOVE, and REPLACE
- The ACID types User, DCA, VCA, ZCA, LSCA, and SCA
- ACID(MAINTAIN) authority and MISC4(CERTUSER) FOR user IDs, MISC4(CERTSITE) for CERTSITE ACID, and MISC4(CERTAUTH) for CERTAUTH ACID.

Examples: USAGE keyword

This example indicates that a certificate is a site certificate:

```
TSS ADDTO(USERA) KEYRING(USERARNG)
                 RINGDATA(USERB,USERBCER)
                 USAGE(CERTSITE)
```

This example indicates that a certificate is a certificate authority :

```
TSS ADDTO(USERA) KEYRING(USERARNG)
                 RINGDATA(USERB,USERBCER)
                 USAGE(CERTAUTH)
```

USER Keyword—Maintain Resource Access

Valid on z/OS and z/VM.

Use the USER keyword to grant or remove access to unownable, installation-defined resources.

Customize the application programs to call CA Top Secret to verify access to installation-defined resources. You can define up to 40 unowned resource classes.

This keyword has the following format:

TSS ADD(*acid*) USER(*class*,*value*)

class

One character code alpha, numeric or national character. The value is tested using RACROUTE.

value

1 to 8 character class-value

This keyword is used with:

- The commands CREATE ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ALL Record
- MISC1(USER) authority

Examples: USER keyword

This example allows a user to access several unowned user resources:

```
TSS ADDTO(GENU8) USER(U,WELL)
```

This example removes access to unowned installation-defined resources:

```
TSS REMOVE(GENU8) USER(U,WELL)
```

USERNL1 and USERNL2 Keywords—CTS National Language

Valid on z/OS

Use the USERNL1 and USERNL2 keywords to set the National Language code for CTS sign ons. USERNL1 is the primary and USERNL2 is the secondary.

This keyword has the following format:

TSS ADDTO(*acid*) [USERNL1(*code*)|USERNL2(*code*)]

code

For information on valid language codes, see the CTS documentation.

Prefix Length: Three characters.

Capacity of List: One per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, and WHOHAS
- ACID(MAINTAIN) authority

USING Keyword—Specify a Model ACID

Valid on z/OS, z/VSE, and z/VM.

Use the USING keyword to specify an existing ACID to use as a model when you are performing the following activities:

- Creating a new ACID based on the specified ACID model.
- Modeling an existing ACID after the specified ACID model.

For a CREATE command, the model ACID information that is available from a TSS LIST(*modelacid*) DATA(BASIC,TSO) command is copied to the ACID, along with the INSTDATA, TSO DFLT, SMS, CICS, OMVS, and WORKATTR DATA information.

Note: CA Top Secret also copies the password information; however, CA Top Secret does *not* copy any PHRASE that is assigned to the ACID. Information that is not copied by CREATE USING (such as PERMITs) must be explicitly granted to the new ACID after the CREATE. To determine which information you need to grant, use the COMPARE command to compare the two ACIDs.

The MODEL command produces TSS commands that contain the user ID specified by the ACID keyword, with resources and permissions from the user ID specified in the USING keyword.

This keyword has the following format:

```
TSS CREATE(newacid) USING(model_acid)
```

```
TSS MODEL ACID(new_acid) USING(model_acid) INTO(dataset_name(member_name))
```

model_acid

Specifies the acid whose information will be copied.

new_acid

Specifies the ACID that will be modeled after the ACID specified in the USING keyword. For a MODEL function, you can specify a new ACID name or an existing ACID name. If you specify an existing ACID name, and you want to use the TSS commands produced by the MODEL commands without changing them, you must delete or rename the existing ACID before executing the TSS commands.

***dataset_name*(*member_name*)**

Specifies the data set name and member into which CA Top Secret generates MODEL command output for the new ACID.

This keyword can be used with:

- The commands COMPARE, CREATE, and MODEL
- The ACID types User, Profile, DEPARTMENT, ZONE, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority and DATA(ALL) (for a MODEL function)

  Note: For a CREATE function, all rules of scope and administrative authority are supported. If the model ACID is outside the scope of the administrator (or has an information field that requires authority that the administrator does not have), the CREATE function fails.

Example: Create an ACID Based On an Existing ACID

This example creates an ACID (USER02) that is modeled after existing ACID USER01 (including USER01's name and password):

```
TSS CREATE(USER02) USING(USER01)
```

Example: Create an ACID with an Assigned Name and Password

This example gives USER02 the name John Smith and initial password SMIJO01. Passwords expire every seven days.

```
TSS CREATE(USER02) USING(USER01)
                   NAME('JOHN SMITH')
                   PASSWORD(SMIJO01,7,EXP)
```

Example: Omit an Information Field from a New ACID

To omit an information field from the new ACID, enter the keyword with no value specified in the parentheses. This example omits the INSTDATA keyword from USER02:

```
TSS CREATE(USER02) USING(USER01)
                   INSTDATA()
```

All ACID attributes in USER01 not desired in USER02 must be explicitly removed from USER02 after the CREATE. ACID attributes (SUSPEND and NORESCHK) have no values to nullify with () string values and therefore cannot be nullified within the CREATE syntax.

Example: Model an Existing ACID After a Model ACID

This example generates the model command output for new ACID NewRac (by using existing ACID Rachael) and places the output in MVS PDS data set HORKR02.RACHAEL.MODEL without a member name. The member will be set to the ACID name in the ACID keyword NewRac:

```
TSS MODEL USING(Rachael) ACID(NewRac) INTO(horkr02.rachael.model)
```

More information:

COMPARE Function—Compare ACIDs (see page 45)

USING Keyword with COMPARE—Specify ACID to Compare

Valid on z/OS, z/VSE, and z/VM.

Specify the USING keyword on a COMPARE command to identify an ACID to compare to the ACID that is specified in the COMPARE command.

This keyword has the following format:

```
TSS COMPARE(acid1) USING(acid2)
```

acid1

Specifies the first ACID to compare.

acid2

Specifies the second ACID to compare.

This keyword is used with:

- The commands CREATE, COMPARE, and MODEL
- ACID(MAINTAIN) authority and DATA(ALL)

Example: Compare the Security Records of ACIDs Rachel01 and Rachel02

This example compares the security records of ACIDs rachel01 and rachel02:

```
TSS COMPARE(rachel01) USING(rachel02)
```

More information:

COMPARE Function—Compare ACIDs (see page 45)

VMUSER Keyword—User Authority

Valid on z/VM.

Use the VMUSER keyword to limit specific user IDs to the range of authority being permitted. This is only valid with CP commands and diagnose codes.

Note: The VMUSER keyword is invalid for the REVOKE command. To REVOKE a PERMIT containing a VMUSER, REVOKE access for the CPCMD or diagnose without using the VMUSER keyword. To change the VMUSER data on a PERMIT, REVOKE access without using the VMUSER keyword, and then REVOKE the PERMIT with the new VMUSER data.

This keyword has the following format:

```
TSS PERMIT(acid) CPCMD(prefix(s))
                 VMUSER(userid,...)
```

Example: VMUSER keyword

This example permits a Systems Programmer to use the spooling command CHANGE for files belonging to USER01:

```
TSS PERMIT(SYSPROG) CPCMD(CHANGE)
                    VMUSER(USER01)
```

VSECATBT Keyword—Catalog B-transients Rights

Valid on z/VSE.

Use the VSECATBT keyword to assign the z/VSE special right to catalog B-transients. This attribute corresponds to the RIGHT attribute of the DTSECTAB.

This keyword has the following format:

```
TSS ADDTO(acid) VSECATBT
```

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, and REPLACE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: VSECATBT keyword

This example assigns the z/VSE special right to catalog B-transients:

```
TSS ADDTO(SYSA) VSECATBT
```

This example removes the above attribute:

```
TSS REMOVE(SYSA) VSECATBT
```

VSEMCON Keyword—Master Console Rights

Valid on z/VSE.

Use the VSEMCON keyword to assign the z/VSE special right to open the master console. This attribute corresponds to the MCON attribute of the DTSECTAB.

This keyword has the following format:

```
TSS ADDTO(acid) VSEMCON
```

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, and REPLACE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: VSEMCON keyword

This example assigns the z/VSE special right to open the master console:

```
TSS ADDTO(SYSA) VSEMCON
```

This example removes the above attribute:

```
TSS REMOVE(SYSA) VSEMCON
```

VSERDD Keyword—Read Directory Rights

Valid on z/VSE.

Use the VSERDD keyword to assign the z/VSE special right to read directories. This attribute corresponds to the READDIR attribute of the DTSECTAB.

This keyword has the following format:

```
TSS ADDTO(acid) VSERDD
```

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, and REPLACE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: VSERDD keyword

This example assigns the z/VSE special right to read directories:

```
TSS ADDTO(SYSA) VSERDD
```

This example removes the above attribute:

```
TSS REMOVE(SYSA) VSERDD
```

VSESYSAD Keyword—Security Administrator Rights

Valid on z/VSE.

Use the VSESYSAD keyword to assign the z/VSE special right of security administrator. This attribute corresponds to the AUTH attribute of the DTSECTAB.

This keyword has the following format:

```
TSS ADDTO(acid) VSESYSAD
```

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, and REPLACE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(MAINTAIN) authority

Examples: VSESYSAD keyword

This example assigns the z/VSE special right of security administrator:

```
TSS ADDTO(SYSA) VSESYSAD
```

This example removes the above attribute:

```
TSS REMOVE(SYSA) VSESYSAD
```

VSUSPEND Keyword—Remove ACID Suspension

Valid on z/OS and z/VM.

Use the VSUSPEND keyword to remove the suspension of an ACID suspended for access violation reasons. Only the the MSCA can issue a command with this keyword.

This keyword has the following format:

```
TSS REMOVE(acid) VSUSPEND
```

This keyword is used with:

- The command REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC1(SUSPEND) authority
- Administrators who do not have MISC1(SUSPEND) authority can remove a VSUSPEND from users within their scope if given UPDATE access to entity TSSCMD.USER.REMOVE.VSUSPEND in the CASECAUT resource class

Examples: VSUSPEND keyword

This example removes the VSUSPEND attribute from an ACID:

```
TSS REMOVE(ACARP) VSUSPEND
```

This example remove suspend command will also remove VSUSPEND:

```
TSS REMOVE(ACARP) SUSPEND
```

WAACCNT Keyword—APPC Processing Account Number

Valid on z/OS, ESA 4.2.0 and above only.

Use the WAACCNT keyword to provide an account number for z/OS APPC processing.

The value specified for WAACCNT overrides an account number specified on the job statement.

Note: In order for this information to be extracted from the ACID's User Record, TAILOR_ACCOUNT and TAILOR_SYSOUT must be set to YES for the appropriate Transaction Program (TP).

This keyword has the following format:

```
TSS ADDTO(acid) WAACCNT(accntname)
```

accntname

Specifies the account number. If spaces are used contain the value within single quotes.

Range: Up to 255 characters

This keyword is used with:

- The commands ADDTO, CREATE, REPLACE, REMOVE, and WHOHAS
- The ACID types MSCA, SCA, LSCA, ZCA, VCA, DCA, and User
- MISC2(WORKATTR) authority

Examples: WAACCNT keyword

This example generates SYSOUT information USER01 under account number abc123:

```
TSS ADDTO(USER01) WAACCNY(abc123)
```

WAADDRn Keyword—Additional SYSOUT Information

Valid on z/OS, ESA 4.2.0 and above.

Use the WAADDRn keyword to indicate up to four additional lines of SYSOUT delivery information.

Note: In order for this information to be extracted from the ACID's User Record, TAILOR_ACCOUNT and TAILOR_SYSOUT must be set to YES for the appropriate Transaction Program (TP).

This keyword has the following format:

```
TSS ADDTO(acid) WAADDR1(info)
                WAADDR2(info)
                WAADDR3(info)
                WAADDR4(info)
```

info

Delivery information. If spaces are used contain the value within single quotes (for example, 'Immediate Delivery').

Range: Up to 60 characters

This keyword is used with:

- The commands ADDTO, CREATE, REPLACE, REMOVE, and WHOHAS
- The ACID types SCA, LSCA, ZCA, VCA, DCA, and User
- MISC2(WORKATTR) authority to administer the WAADDRn
- DATA(WORKATTR) authority to list WAADDRn information

Example: WAADDRn keyword

This example delivers SYSOUT information generated by USER01 to Sue Brown in room 553 of the Downtown Corporate Office Building in Chicago:

```
TSS ADDTO(USER01) WANAME('Sue Brown')
                  WABLDG('Downtown Corporate Bldg')
                  WAADDR1(Chicago)
                  WAROOM(553)
```

WABLDG Keyword—SYSOUT Building Delivery

Valid on z/OS, ESA 4.2.0 and above.

Use the WABLDG keyword to indicate the building SYSOUT information is delivered to.

Note: For this information to be extracted from the ACID's User Record, TAILOR_ACCOUNT and TAILOR_SYSOUT must be set to YES for the appropriate Transaction Program (TP).

This keyword has the following format:

```
TSS ADDTO(acid) WABLDG(name)
```

name

Specifies the building name. If spaces are used contain the value within single quotes (for example, 'Corporate Bldg. East').

Range: Up to 60 characters

This keyword is used with:

- The commands ADDTO, CREATE, REPLACE, REMOVE, and WHOHAS
- The ACID types SCA, LSCA, ZCA, VCA, DCA, and User
- MISC2(WORKATTR) authority

Examples: WABLDG keyword

This example designates that SYSOUT information generated by USER01 is delivered to Janet Adams at the downtown corporate office:

```
TSS ADDTO(USER01) WANAME('Janet Adams')
                  WABLDG('Corp Office Downtown')
```

WADEPT Keyword—SYSOUT Department Delivery

Valid on z/OS, ESA 4.2.0 and above.

Use the WADEPT keyword to indicate the department SYSOUT information is delivered to.

Note: For this information to be extracted from the ACID's User Record, TAILOR_ACCOUNT and TAILOR_SYSOUT must be set to YES for the appropriate Transaction Program (TP).

This keyword has the following format:

```
TSS ADDTO(acid) WADEPT(deptname)
```

deptname

Specifies the department name. If spaces are used contain the value within single quotes (for example, 'Accounts Payable Department').

Length: Up to 60 characters

This keyword is used with:

- The commands ADDTO, CREATE, REPLACE, REMOVE, WHOHAS
- The ACID types SCA, LSCA, ZCA, VCA, DCA, and User.
- MISC2(WORKATTR) authority to administer WADEPT
- DATA(WORKATTR) authority to list WADEPT information

Example: WADEPT keyword

This example designates that SYSOUT information generated by USER01 is delivered to the Accounting Department:

```
TSS ADDTO(USER01) WADEPT('Accounting Department')
```

WAIT Keyword—Synchronous or Asynchronous Processing

Valid on z/OS and z/VM.

Use the WAIT keyword to indicate whether a CA Top Secret command is processed synchronously (locked until a response is received from the targeted node) or asynchronously.

This keyword has the following format:

```
TSS function(acid) keyword(s)
                   WAIT(YES|NO)
```

YES

Specifies synchronous processing (CPF waits for a response from all remote nodes before processing resumes). When this type of processing is in effect, the CPF Recovery File is not used to save transmitted commands.

NO

Specifies asynchronous processing (CPF does not wait for a response). A response is returned to the CPF Journal File for that node only. The terminal keyboard is freed up as soon as the command is accepted. When this processing is in effect, the CPF Recovery File saves transmitted commands. Commands targeted for the local machine only are not saved.

This keyword is used with:

- The commands CREATE, ADDTO, LIST, REMOVE, PERMIT, REVOKE, REPLACE, RENAME, ADMIN, DEADMIN, WHOOWNS, WHOHAS, MOVE, and DELETE
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- No explicit authority

Examples: WAIT keyword

This example adds data sets beginning with SYS1. to the Accounting Department, transmits them to all nodes whose names start with R and processes them synchronously:

```
TSS ADDTO(ACCDPT) DSNAME(SYS1.)
                  TARGET(R*)
                  WAIT(Y)
```

This example removes the data sets:

```
TSS REMOVE(ACCDPT) DSNAME(SYS1.)
                   TARGET(R*)
                   WAIT(Y)
```

WANAME Keyword—SYSOUT Person Delivery

Valid on z/OS, ESA 4.2.0 and above only.

Use the WANAME keyword to indicate who SYSOUT information is delivered to.

Note: In order for this information to be extracted from the ACID's User or Profile Record, TAILOR_ACCOUNT and TAILOR_SYSOUT must be set to YES for the appropriate Transaction Program (TP).

This keyword has the following format:

TSS ADDTO(*acid*) WANAME(*name*)

name

Specifies the user name. If spaces are used, contain the value within single quotes (for example, 'John Smith').

Length: Up to 60 characters

This keyword is used with:

- The commands ADDTO, CREATE, REPLACE, REMOVE, WHOHAS
- The ACID types User and Profile
- MISC2(WORKATTR) authority to administer WANAME
- DATA(WORKATTR) authority to issue a list command for WANAME information

Example: WANAME keyword

This example designates John Smith as the person to whom SYSOUT information generated by USER01 is delivered:

```
TSS ADDTO(USER01) WANAME('John Smith')
```

WAROOM Keyword—SYSOUT Room Delivery

Valid on z/OS, ESA 4.2.0 and above only.

Use the WAROOM keyword to indicate the room SYSOUT information is delivered to.

Note: In order for this information to be extracted from the ACID's User or Profile Record, TAILOR_ACCOUNT and TAILOR_SYSOUT must be set to YES for the appropriate Transaction Program (TP).

This keyword has the following format:

```
TSS ADDTO(acid) WAROOM(roomname)
```

roomname

Specifies the room name. If spaces are used contain the value within single quotes (for example, 'Second Floor Lab').

Length: Up to 60 characters

This keyword is used with:

- The commands ADDTO, CREATE, REPLACE, REMOVE, and WHOHAS
- The ACID types User and Profile
- MISC2(WORKATTR) authority to administer WAROOM
- DATA(WORKATTR) authority to list WORKATTR information

Example: WAROOM keyword

This example specifies that SYSOUT information generated by users assigned to the RESDPROF profile is delivered to room 553:

```
TSS ADDTO(RESDPROF) WAROOM('Room 553')
```

XCOMMAND Keyword—Command Restrictions

Valid on z/OS.

Use the XCOMMAND keyword to *prevent* ACIDs from using a specified command or subset of commands available within that facility.

This keyword has the following format:

```
TSS ADDTO(acid) XCOMMAND(facility,(cmdname[(G)]))
```

For purposes of LCF protection, facility entities USER77 through USER221 are not considered unique.

Prefix length

One to eight characters. Entries are treated as prefixes only if the generic indicator (G) follows the entry.

Capacity of list

One to 30 command names or prefixes per CA Top Secret command.

This keyword is used with:

- The commands CREATE, REPLACE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ALL Record
- MISC1(LCF) authority to ADDTO or REMOVE the XCOMMAND keyword from ACIDs
- MISC9(GLOBAL) authority to ADDTO or REMOVE the XCOMMAND resource class to or from the ALL record

Examples: XCOMMAND keyword

This example allows users attached to PROF01 to execute all TSO ISPF commands except for the SPF panels 3.4 and 3.5:

```
TSS ADDTO(PROF01) XCOMMAND(TSO,(SPF34,SPF35))
```

The letter G next to the command prefix indicates that the ACID is prohibited from using a command beginning with the prefix specified.

This example indicates that user EJ001 cannot use TSO commands prefixed by SPF or any IBM defined SPF panels:

```
TSS ADDTO(EJ001) XCOMMAND(TSO,(SPF(G)))
```

To restrict the use of batch programs:

```
TSS ADDTO(USER07) XCOMMAND(BATCH,(PROG57))
```

To restrict all users from commands used in a specific facility:

```
TSS ADDTO(ALL) XCOMMAND(facility,(commands...))
```

To remove the XCOMMAND restriction:

```
TSS REMOVE(EJ001) XCOMMAND(TSO,(SPF(G)))
```

XSUSPEND Keyword—Suspend ACID

Valid on z/OS and z/VM.

Use the XSUSPEND keyword to suspend an ACID suspended by the installation exit. Only the the MSCA can issue a command with this keyword.

This keyword has the following format:

```
TSS REMOVE(acid) XSUSPEND
```

This keyword is used with:

- The command REMOVE
- The ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- MISC1(SUSPEND) authority
- Administrators who do not have MISC1(SUSPEND) authority can remove an XSUSPEND from users within their scope if given UPDATE access to entity TSSCMD.USER.REMOVE.XSUSPEND in the CASECAUT resource class

Examples: XSUSPEND keyword

This example removes the XSUSPEND attribute from an ACID:

```
TSS REMOVE(ACARP) XSUSPEND
```

This example will also remove XSUSPEND:

```
TSS REMOVE(ACARP) SUSPEND
```

XTRANSACTIONS Keyword—ACID Transaction Restriction

Valid on z/OS and z/VSE.

Use the XTRANSACTIONS keyword to *restrict* ACIDs from using a specific transaction, or subset of the transactions, available within that facility.

This keyword has the following format:

```
TSS ADDTO(acid) XTRANSACTIONS(facility,(transactions[(G)]))
```

For purposes of LCF protection, facility entities USER77 through USER221 are not considered unique.

Prefix length

One to eight characters. Entries are treated as a prefix only if the generic indicator (G) follows the entry.

Capacity of list

One to 30 transaction names or prefixes per TSS command.

This keyword is used with:

- The commands CREATE REPLACE, ADDTO, and REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ALL Record
- MISC1(LCF) authority to ADDTO or REMOVE the XTRANSACTIONS keyword from ACIDs
- MISC9(GLOBAL) authority to ADDTO or REMOVE the XTRANSACTIONS resource class to or from the ALL record.

Examples: XTRANSACTIONS keyword

This example prohibits a group of CICS clerks from using a set of transactions:

```
TSS ADDTO(PROF1) XTRANSACTIONS(CICSP,(CP01,CP02,CP03))
```

The letter G next to the transaction prefix indicates that the user is prohibited from using transactions that begin with the prefix specified.

This example indicates that user EJ001 is prohibited from using transactions prefixed by CPO.

```
TSS ADDTO(EJ001) XTRANSACTIONS(CICSP,(CPO(G)))
```

This example removes the XTRANSACTIONS restriction:

```
TSS REMOVE(EJ001) XTRANSACTIONS(CICSP,(CPO(G)))
```

YEAR Keyword—Calendar Year

Valid on z/OS, z/VSE, and z/VM.

Use the YEAR keyword to add, remove, replace or list a year field in the calendar record of the SDT Record.

This keyword has the following format:

```
TSS ADD(SDT) CALENDAR(cal-name)
             [YEAR(yyyy)]
             [DAYS(days)]
             [INCLUDE(mm/dd,...]
             [EXCLUDE(mm/dd,...)]
```

yyyy

Specifies a year in which to apply a calendar record.

Default: The current year

This keyword is used with:

- The commands ADDTO, REMOVE, REPLACE, LIST, and DELETE
- The SDT Record exclusively
- MISC3(SDT) authority

Example: YEAR keyword

This example creates a 2006 calendar named FIN06 for the first quarter financial month close:

```
TSS ADDTO(SDT) CALENDAR(FIN06)
               YEAR(2006)
               INCLUDE(01/30,01/31,02/28,02/29,03/30,03/31)
```

ZONE Keyword—Zone ACIDs

Valid on z/OS, z/VSE, and z/VM.

Use the ZONE keyword to:

- Assign an ACID to a zone
- List data about ACIDs in a specific zone

CA Top Secret automatically assigns the new ACID to the ZCA's zone. The ZONE keyword is only required when the person entering the command is an SCA.

CA Top Secret only displays data concerning ACIDs within the administrator's scope. An MSCA, an authorized SCA or an authorized LSCA, could obtain LIST data for the entire site. A ZCA cannot specify the ZONE keyword.

This keyword has the following format:

```
TSS CREATE(divacid|ZCAacid) TYPE(DIVISION|ZCA)
                            NAME('ZCA or Division name')
                            ZONE(acid)

TSS LIST(ACID|ACIDS) DATA(datatype(s))
                     TYPE(acidtype)
                     ZONE(acid)
```

This keyword is used with:

- The commands LIST, MOVE, and CREATE
- The ACID types Division and ZCA

Example: ZONE keyword

This example creates a new Accounting Division placed in the Parts Zone:

```
TSS CREATE(ACCT01) NAME('ACCT DIV')
                   TYPE(DIVISION)
                   ZONE(PARTZON)
```

Note: To add a new department to a zone add that department to a division. Departments cannot be directly added to a zone.

This example lists the names of all users in the TECH Zone:

```
TSS LIST(ACIDS) DATA(NAMES)
                ZONE(TECH)
```

Chapter 4: Resource Classes

This section contains the following topics:

About Resource Classes (see page 553)
ABSTRACT Resource Class—Secure Resource Classes (see page 554)
ACID Resource Class—Secure ACIDS (see page 556)
APPCLU Resource Class—Secure APPCLU Links (see page 557)
APPCPORT Resource Class—Specify VTAM LU Name (see page 559)
APPCSI Resource Class—Secure APPC Side Information Files (see page 561)
APPCTP Resource Class—Secure APPC Transaction Programs (see page 563)
APPLICATION Resource Class—Secure IMS Application Group Names (see page 565)
AREA Resource Class—Secure CA-IDMS Database Areas (see page 566)
CAADMIN Resource Class—Secure CAADMIN Administration (see page 568)
CACMGR Resource Class-Secure CACMGR (see page 570)
CACCFDSN Resource Class—Secure CACCFDSN Data Sets (see page 572)
CACCFMEM Resource Class—Secure CACCFMEM Members (see page 574)
CACMD Resource Class—Secure CA Commands (see page 576)
CAGSVX Resource Class—Secure CAGSVX for SYSVIEW (see page 578)
CALIBMEM Resource Class—Secure CALIBMEM Members (see page 580)
CAREPORT Resource Class—Secure Reports (see page 582)
CASECAUT Resource Class—Authorize Restricted Administrative Priviliges (see page 584)
CATAPE Resource Class—Secure Tape Volumes (see page 586)
CAVAPPL Resource Class—Secure CAVAPPL for CAVMAN (see page 588)
CBIND Resource Class—Secure CBIND for z/OS Objects (see page 590)
CIMS Resource Class—Secure IMS Commands (see page 592)
CPCMD Resource Class—Secure CP Commands (see page 594)
CPU Resource Class—Secure CPU Access (see page 596)
CSFKEYS Resource Class—Secure ICSF CSFKEYS (see page 598)
CSFSERV Resource Class—Secure ICSF CSFSERV (see page 600)
CTSDB2 Resource Class—Secure CTS DB2ENTRY and DB2TRAN Resources (see page 602)
DATABASE Resource Class—Secure Databases (see page 603)
DB2 Resource Class—Secure DB2 Databases (see page 605)
DB2BUFFP Resource Class—Secure DB2 Buffer Pools (see page 608)
DB2COLL Resource Class—Secure DB2 Collection ID (see page 610)
DB2DBASE Resource Class—Secure DB2 Databases (see page 612)
DB2PKG Resource Class—Secure DB2 Packages (see page 614)
DB2PLAN Resource Class—Secure DB2 Plans (see page 616)
DB2SEQ Resource Class—Identify DB2 Sequences (see page 618)
DB2STOGP Resource Class—Secure DB2 Storage Groups (see page 620)
DB2SYS Resource Class—Secure DB2 Privileges and Authorities (see page 622)
DB2TABLE Resource Class—Secure DB2 Tables (see page 624)
DB2TABSP Resource Class—Secure DB2 Table Spaces (see page 626)
DBD Resource Class—Secure IMS Database Name (see page 628)
DCSS Resource Class—Secure Segments (see page 630)
DCT Resource Class—Secure CICS DCT Data (see page 632)
DCTABLE Resource Class—Secure CA SYSVIEW DCTABLE (see page 634)
DEVICES Resource Class—Secure Devices (see page 636)
DFTABLE Resource Class—Secure CA SYSVIEW DFTABLE (see page 638)
DIAGNOSE Resource Class—Secure CP Diagnose Codes (see page 640)
DIRECTRY Resource Class—Secure DIRECTRY for TSSVM (see page 642)

DLFCLASS Resource Class—Determine DLF Record Access (see page 644)
DRTABLE Resource Class—Secure CA SYSVIEW DRTABLE (see page 646)
DSNAME Resource Class—Secure Data Sets (see page 648)
DSPACE Resource Class—Restrict Access to Databases (see page 650)
DSTABLE Resource Class—Secure CASYSVIEW DSTABLE (see page 652)
DTADMIN Resource Class—Secure CASYSVIEW DTADMIN (see page 654)
DTSYSTEM Resource Class—Secure CASYSVIEW DTSYSTEM (see page 656)
DTTABLE Resource Class—Secure CASYSVIEW DTTABLE (see page 658)
DTUTIL Resource Class—Secure CASYSVIEW DTUTIL (see page 660)
DXTABLE Resource Class—Secure CASYSVIEW DXTABLE (see page 662)
EJBROLE Resource Class—Secure z/OS Enterprise Beans (see page 664)
FCT Resource Class—Secure CICS File Control Table Entries (see page 666)
FIELD Resource Class—Secure Database Fields (see page 668)
FSACCESS Resource Class—Secure zFS File System Container (see page 670)
GEJBROLE Resource Class—Secure z/OS Enterprise Beans (see page 672)
GSOMDOBJ Resource Class—Secure z/OS GSOMDOBJ Objects (see page 674)
HFSSEC Resource Class—Secure HFS Files (see page 676)
IBMFAC Resource Class—Determine IBM Facilities Ownership (see page 678)
IBMGROUP Resource Class—Assign Group Names (see page 680)
IUCV Resource Class—Secure IUCV Target Users (see page 682)
JCT Resource Class—Secure JCT Entries (see page 684)
JESINPUT Resource Class—Secure JES Job Entry Sources (see page 686)
JESJOBS Resource Class—Secure Jobs (see page 688)
JESSPOOL Resource Class—Secure JES Spool Data Sets (see page 690)
JOBNAME Resource Class—Secure Jobname (see page 692)
LDAP Resource Class—Secure LDAP Administration (see page 694)
LFSCLASS Resource Class—Secure MLF LFSCLASS (see page 696)
LOGSTRM Resource Class—Secure System Logger (see page 698)
MGMTCLAS Resource Class—Secure Management Classes (see page 699)
MQADMIN Resource Class—Secure MQM Commands (see page 701)
MQCMDS Resource Class—Secure MQM Commands (see page 703)
MQCONN Resource Class—Secure MQM Connections (see page 705)
MQNLIST Resource Class—Secure MQM Name Lists (see page 707)
MQPROC Resource Class—Secure MQM Processes (see page 709)
MQQUEUE Resource Class—Secure the MQM Queue (see page 711)
MXADMIN Resource Class—Secure MXM Commands (see page 713)
MXNLIST Resource Class—Secure MXM Name Lists (see page 715)
MXPROC Resource Class—Secure MXM Topics (see page 717)
MXQUEUE Resource Class—Secure the MXM Queue (see page 719)
MXTOPIC Resource Class—Secure MXM Topics (see page 721)
NETCMDS Resource Class—Secure NETVIEW NETCMDS (see page 723)
NETSPAN Resource Class—Secure NETVIEW NETSPAN (see page 725)
NODES Resource Class— Control NJE Jobs and SYSOUT (see page 727)
OPCMD Resource Class—Secure SVC 34 Commands (see page 729)
OPERCMDS Resource Class—Secure JES Operator Commands (see page 731)
OTRAN Resource Class—Secure Ownable Transactions (see page 733)
PANAPT Resource Class—Secure PANAPT for PANVELT (see page 735)
PANEL Resource Class—Secure Panels (see page 737)
PDSMEMn Resource Class—Secure Partitioned Data Set Members (see page 739)

PKISERV Resource Class—Control Access to R_Pkiserv Administrative Functions (see page 740)
PPT Resource Class—Secure the CICS PPT (see page 742)
PROGRAM Resource Class—Secure Programs and Utilities (see page 744)
PROPCNTL Resource Class—Secure Special ACIDs (see page 746)
PSB Resource Class—Secure the IMS Program Specification Block (see page 747)
PSFMPL Resource Class—Secure Output Labeling Suppression (see page 749)
PTKTDATA Resource Class—PassTicket Data Resources (see page 751)
RECIPID Resource Class—Secure CADISPATCH RECIPID (see page 752)
RESLIST Resource Class—Secure TSSAI RESLIST (see page 754)
RODMMGR Resource Class—Secure NETVIEW RODMMGR (see page 756)
ROSRES Resource Class—Determine CA Roscoe CA—Vollie Command Ownership (see page 758)
SCHEDULE Resource Class—Secure SCHEDULE for CA7 (see page 760)
SDSF Resource Class—Secure SDSF Commands (see page 762)
SERVAUTH Resource Class—Secure TCP/IP Resources (see page 764)
SERVER Resource Class—Secure z/OS SERVER Objects (see page 766)
SMESSAGE Resource Class—Secure TSO Messages (see page 768)
SOMDOBJS Resource Class—Secure z/OS SOMDOBJS Objects (see page 770)
SPI Resource Class—Secure CEMT and CICS (see page 772)
STATION Resource Class—Secure CADISPATCH and CASCHEDULER (see page 781)
STORCLAS Resource Class—Secure SMS Storage Classes (see page 783)
SUBSCHEM Resource Class—Secure CAIDMS Subschema Names (see page 785)
SURROGAT Resource Class—Restrict Preset Security (see page 787)
SYSAUTO Resource Class—IBM Automation Control for z/OS Resources (see page 788)
SYSCONS Resource Class—Secure IBM Consoles (see page 790)
SYSMVIEW Resource Class—Secure SystemView Segments (see page 792)
TCICSTRN Resource Class—Secure CICS TCICSTRN RESCLASS (see page 794)
TERMINAL Resource Class—Secure Terminals (see page 796)
TIMS Resource Class—Secure IMS TIMS (see page 800)
TOTAL Resource Class—Secure TOTAL for CINCOM (see page 802)
TSOACCT Resource Class—Secure TSO Logon Account Codes (see page 804)
TSOAUTH Resource Class—Secure TSO User Attributes (see page 806)
TSOPRFG Resource Class—Secure TSO Performance Groups (see page 808)
TSOPROC Resource Class—Secure TSOP Logon Procs (see page 810)
TST Resource Class—Secure CICs Temporary Storage Table Names (see page 812)
UR1/UR2 Resource Class—Secure Resource Restrictions (see page 813)
USRCLASS Resource Class—Secure Resource Classes (see page 816)
VMANAPPL Resource Class—Secure CAVMAN VMANAPPL (see page 818)
VMCF Resource Class—Secure VMCF Communication Targets (see page 820)
VMDEV Resource Class—Secure z/VM Real Devices (see page 822)
VMDIAL Resource Class—Secure Dial Access Virtual Machines (see page 824)
VMMACH Resource Class—Authorize AUTOLOG Users (see page 826)
VMMDISK Resource Class—Secure Minidisks (see page 828)
VMNODE Resource Class—Secure z/VM Nodes (see page 831)
VMRDR Resource Class—Secure Virtual Machine Readers (see page 833)
VOLUME Resource Class—Secure DASD and Tape (see page 835)
VSELIB Resource Class—Secure TSSVSE Libraries (see page 837)
VSEPART Resource Class—Secure Partitions (see page 839)

VSEUSER Resource Class—Secure TSSVSE Users (see page 841)
VTAMAPPL Resource Class—Secure VTAM ACB Access (see page 843)
WRITER Resource Class—Secure Job Output (see page 845)
XCSFKEY Resource Class—Secure Exportation of ICSF Cryptographic Keys (see page 847)
ZMFAPLA Resource Class—Secure z/OSMF Authorization Roles (see page 849)

About Resource Classes

All resources in this chapter (with the exception of data sets, minidisks, DB2 databases, DB2 tables, and DB2 table spaces) honor the NONGENERIC attribute which can protect a resource by its fully qualified name rather than its prefix.

ABSTRACT Resource Class—Secure Resource Classes

Valid on z/OS and z/VSE.

Use ABSTRACT to secure installation-defined resource classes.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) ABSTRACT(`*`oper`*`,...)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) ABSTRACT(`*`prefix(es)`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

ABSTRACT is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- ABSTRACT(OWN) authority to ADD or REMOVE ownership of ABSTRACT resources from ACIDs
- ABSTRACT(XAUTH) authority to PERMIT or REVOKE access to ABSTRACT resources within their scope

The administrator can use Expiration, Facility, Program Pathing, Time/Day, and Actions methods to control access to ABSTRACT resources.

Examples: ABSTRACT resource class

This example protects linkage editor setcode by using the CA Top Secret reorganized name of AC1. AC1 is owned by the Corporate Department:

```
TSS ADDTO(CORPORAT) ABSTRACT(AC1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) ABSTRACT(AC1)
```

This example permits users in the Technical Services Department to access AC1 on Fridays only:

```
TSS PERMIT(TECHPROF) ABSTRACT(AC1) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access XDT98000:

```
TSS PERMIT(TECHPROF) ABSTRACT(XDT98000)
```

This example revoke access:

```
TSS REVOKE(TECHPROF) ABSTRACT(AC1)
```

ACID Resource Class—Secure ACIDS

Valid on z/OS, z/VSE, and z/VM.

Use ACID to secure special Accessor IDs, such as those used to submit jobs.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) ACID(acid)
```

acid

The acid being secured.

Length: 1 to 8 characters

Capacity: 1 to 5 ACIDs per command

ACID is used with:

- The commands PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- ACID(XAUTH) authority to PERMIT or REVOKE ACIDs

The administrator can use Expiration, Facility, Program Pathing, Time/Day, and Actions methods to control the use of job submission ACIDs.

Examples: ACID resource class

This example permits USERA to submit a batch job with a USER= of USERB:

```
TSS PERMIT(USERA) ACID(USERB)
```

This example revokes USERA's ability to code USER=USERB on a job statement:

```
TSS REVOKE(USERA) ACID(USERB)
```

APPCLU Resource Class—Secure APPCLU Links

Valid on z/OS and z/VSE.

Use APPCLU to secure APPCLU links.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) APPCLU(`*`xxxxx`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) APPCLU(`*`xxxxx`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command.

APPCLU is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- APPCLU (OWN) authority to ADD or REMOVE ownership of APPCLU resources from ACIDs
- APPCLU(XAUTH) authority to PERMIT or REVOKE access to APPCLU resources

The administrator can use Expiration, Facility, Program Pathing, Time/Day, and Actions methods to control access to APPCLU resources.

Examples: APPCLU resource class

This example protects linkage editor setcode by using the CA Top Secret reorganized name of AC1. AC1 is owned by the Corporate Department:

```
TSS ADDTO(CORPORAT) APPCLU(AC1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) APPCLU(AC1)
```

This example permits users in the Technical Services Department to access AC1 on Fridays only:

```
TSS PERMIT(TECHPROF) APPCLU(AC1)
                     DAYS(FRIDAY)

TSS PERMIT(TECHPROF) APPCLU(XDT98000)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) APPCLU(AC1)
```

APPCPORT Resource Class—Specify VTAM LU Name

Valid on z/OS, ESA 4.2.0 and above only.

Use APPCPORT to indicate the VTAM LU name of the Logical Unit (LU) from which an APPC conversation request must originate. The APPL resource class is used to indicate to which LU a conversation request is made.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) APPCPORT(`*`luname`*`,...)`

Prefix length

One to eight bytes

Capacity of list

One to five LU names per TSS command

Prefixing and masking are both supported.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) APPCPORT(`*`luname`*`...)`

Prefix length

One to eight bytes

Capacity of list

One to five LU names per TSS command

APPCPORT is used with:

- The commands ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA,and MSCA when used with TSS PERMIT/REVOKE
- APPCPORT(OWN) authority to ADD or REMOVE ownership of APPCPORT resources from ACIDs
- APPCPORT(XAUTH) authority to PERMIT or REVOKE access to APPCPORT resources

The administrator can use any of the following methods to control access to APPCPORT resources: Expiration, Program Pathing, Time/Day, and Actions.

Examples: APPCPORT resource class

This example restricts ACIDs belonging to the PROCDEPT so that they can only initiate a Transaction Program (TP) from LU01:

```
TSS ADDTO(PROCDEPT) APPCPORT(LU01)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(PROCDEPT) APPCPORT(LU01)
```

This example permits users with the Technical Services Profile to access LU01 on Fridays only:

```
TSS PERMIT(TECHPROF) APPCPORT(LU01) DAYS(FRIDAY)
```

APPCSI Resource Class—Secure APPC Side Information Files

Valid on z/OS, ESA 4.2.0 and above.

Use APPCSI to determine who can administer the APPC side information files. These files contain the symbolic destination node names used when one Transaction Program (TP) converses with another TP on a different node.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) APPCSI(dbtoken.SYS1.symbolic)
```

dbtoken

One- to eight-character database token associated with the side information file.

SYS1

Indicates that this APPCSI is available to all users on that node. A particular ACID can also be substituted for SYS1.

symbolic

One- to eight-character symbolic destination name associated with the side information file.

In order to view the side information, the administrator requires READ access. In order to create, modify or delete these files, the administrator requires UPDATE access.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) APPCSI(dbtoken.SYS1.symbolic) ACCESS(access)
```

In order to view the side information, the administrator requires READ access. In order to create, modify or delete these files, the administrator requires UPDATE access.

APPCSI is used with:

- The commands ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- APPCSI(OWN) authority to ADD or REMOVE ownership of APPCSI resources from ACIDs
- APPCSI(XAUTH) authority to PERMIT or REVOKE access to APPCSI resources

The administrator can use any of the following methods to control access to APPCSI resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Masking

Prefixing and masking are both supported.

Examples: APPCSI resource class

This example makes LSCA01 responsible for maintaining the side information file for the local system:

```
TSS ADDTO(LSCA01) APPCSI(TOKENA.SYS1.SYMDES1)
```

This example permits users in the Technical Services Department to view the APPCSI for the local system:

```
TSS PERMIT(TECHPROF) APPCSI(TOKENA.SYS1.SYMDES1)
                     ACCESS(READ)
```

APPCTP Resource Class—Secure APPC Transaction Programs

Valid on z/OS, ESA 4.2.0 and above only.

Use APPCTP to secure APPC transaction programs (TPs) and TP profiles. An access level of READ or UPDATE allows the ACID to maintain the TP profile data set (which contains the TP class and JCL required to run the TP) while an access level of EXECUTE allows the ACID to run the TP.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) APPCTP(`*`dbtoken.level.tpname`*`)`

Capacity of list

One to five names per TSS command

dbtoken

One- to- eight character database token associated with the TP profile.

level

Represents one of the following:

SYS1

If the TP is available to all users who can access the LU.

groupid

If the TP is only available to a particular group.

acid

If the TP is only available to a particular user.

tpname

One- to -64 character TP name.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) APPCTP(`*`dbtoken.level.tpname`*`)`

Capacity of list

One to five names per TSS command.

APPCTP is used with:

- The commands ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- APPCTP(OWN) authority to ADD or REMOVE ownership of APPCTP resources from ACIDs
- APPCTP(XAUTH) authority to PERMIT or REVOKE access to APPCTP resources

The administrator can use any of the following methods to control access to APPCTP resources: Expiration, Program Pathing, Time/Day, and Actions.

Masking

Prefixing and masking are both supported.

Examples: APPCTP resource class

The security administrator for APPC resources (LSCA01 in our examples) should create an APPCTP profile for each TP profile, using generic prefixing where appropriate. ACIDs requiring use of the TPs should then be given the appropriate access.

This example makes LSCA01 responsible for maintaining TPB (which is accessible to all users from LU02):

```
TSS ADDTO(LSCA01) APPCTP(TOKENA.SYS1.TPB)
                  ACCESS(UPDATE)
```

This example allows users with the TECHPROF Profile to run TPB:

```
TSS PERMIT(TECHPROF) APPCTP(TOKENA.SYS1.TPB)
                     ACCESS(EXECUTE)
```

APPLICATION Resource Class—Secure IMS Application Group Names

Valid on z/OS.

Use APPLICATION to secure IMS Application Group Names and, when securing APPC conversations (z/OS, ESA 4.2.2 and above), to identify the application name of the LU to which a conversation request is sent.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) APPLICATION(oper,oper...)
```

Prefix length

One to eight characters

Capacity of list

One to five AGNs per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) APPLICATION(prefix(es))
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

APPLICATION is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- APPL(OWN) authority to ADD or REMOVE ownership of APPLICATIONs from ACIDs
- APPLICATION(XAUTH) authority to PERMIT or REVOKE access to APPLICATION resources

The administrator can use any of the following methods to control access to APPLICATIONs: Expiration, Facility, Time/Day, and Actions.

Examples: APPLICATION resource class

This example protects the IMS application 'PAYP' (production payroll), by assigning ownership to the Corporate IMS Department (CORPIMS):

```
TSS ADDTO(CORPIMS) APPLICATION(PAYP)
```

This example removed ownership:

```
TSS REMOVE(CORIMS) APPLICATION(PAYP)
```

This example permits access to the production payroll application on weekdays from 8:00 a.m. to 5:00 p.m.:

```
TSS PERMIT(IP3AA) APPLICATION(PAYP) DAYS(WEEKDAYS) TIMES(08,17)
```

This example revokes access to PAYP:

```
TSS REVOKE(IP3AA) APPLICATION(PAYP)
```

AREA Resource Class—Secure CA-IDMS Database Areas

Valid on z/OS.

Use AREA to secure CA-IDMS database areas.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) AREA(prefix(es))
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) AREA(prefix(es))
                 ACCESS(access levels)
```

Prefix length

One to forty-four characters

Capacity of list

One to five prefixes per TSS command

AREA is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- AREA(OWN) authority to ADD or REMOVE ownership of AREAs from ACIDs
- AREA(XAUTH) authority to PERMIT or REVOKE access to AREAs

The administrator can:

- Specify any or all of the following access levels: NONE, READ, ALL, and UPDATE. If ACCESS is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to AREA resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: AREA resource class

This example gives the Personnel Department ownership of the AREA for the Corporate Personnel Database (PERDB):

```
TSS ADDTO(PERSDP) AREA(PERDB)
```

This example removes ownership:

```
TSS REMOVE(PERSDP) AREA(PERDB)
```

This example permits USER01 to have ALL access to PAYFILE:

```
TSS PERMIT(USER01) AREA(PAYFILE)
                   ACCESS(ALL)
```

This example revokes USER01's access to PAYFILE:

```
TSS REVOKE(USER01) AREA(PAYFILE)
```

CAADMIN Resource Class—Secure CAADMIN Administration

Valid on z/OS.

Use CAADMIN to secure CAADMIN administration.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) CAADMIN(`*`xxxxx`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) CAADMIN(`*`xxxxx`*`)`

Prefix length

One to forty-four characters

Capacity of list

One to five prefixes per TSS command

CAADMIN is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CAADMIN (OWN) authority to ADD or REMOVE ownership of CAADMIN resources from ACIDs
- CAADMIN(XAUTH) authority to PERMIT or REVOKE access to CAADMIN resources

The administrator can use any of the following methods to control access to CAADMIN resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: CAADMIN resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) CAADMIN(XXXXXXXX)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) CAADMIN(XXXXXXXX)
```

This example permits users in the Technical Services Department to access CAADMIN(ABCDEFGH.XXXXXC1) on Fridays only:

```
TSS PERMIT(TECHPROF) CAADMIN(ABCDEFGH.XXXXXC1)
                     DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access to CAADMIN(ABCDEFGH.XXXXXC1):

```
TSS PERMIT(TECHPROF) CAADMIN(ABCDEFGH.XXXXXC1)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) CAADMIN((ABCDEFGH.XXXXXC1)
```

CACMGR Resource Class-Secure CACMGR

Valid on z/OS.

Use the CACMGR resource to secure access to writing and maintaining policies and running reports for CA Compliance Manager.

When used with TSS ADDTO/REMOVE, this resource class has the following format:
`TSS ADDTO(acid) CACMGR(oper,...)`

Prefix length

Two to twenty-six characters.

Capacity of list

One to five data set names or prefixes per TSS command.

Generic prefixing allows the administrator to group a set of similar CACMGR resources, and define them by one generic prefix.

When used with TSS PERMIT/REVOKE, this resource class has the following format:
`TSS PERMIT(acid) CACMGR(prefix(es)) ACCESS(access levels)`

Prefix length

Two to forty-four characters.

Capacity of list

One to five data sets names, prefixes, or masks per TSS command.

CACMGR is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The CACMGR resource ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The CACMGR resource ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CACMGR(OWN) authority to ADD or REMOVE ownership of CACMGR resources from ACIDs

The administrator can:

- Use the following methods to control access to CAMNGR resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.
- Specify the access levels: ALL, NONE, READ, and UPDATE. If you do not specify ACCESS, CA Top Secret defaults to READ.

Examples: CACMGR resource class

This example gives the Inventory Department (INVDEPT) ownership of a CACMGR resource known as COMP.MAN1:

```
TSS ADDTO(INVDEPT) CACMGR(COMP.MAN1)
```

This example removes ownership:

```
TSS REMOVE(INVDEPT) CACMGR(COMP.MAN1)
```

CACCFDSN Resource Class—Secure CACCFDSN Data Sets

Valid on z/OS.

Use CACCFDSN to secure CACCFDSN data sets.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) CACCFDSN(`*`abcdefgh`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) CACCFDSN(`*`abcdefgh`*`.XXXXXX)`

Prefix length

One to forty-four characters

Capacity of list

One to five prefixes per TSS command.

CACCFDSN is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CACCFDSN (OWN) authority to ADD or REMOVE ownership of CACCFDSN resources from ACIDs
- CACCFDSN(XAUTH) authority to PERMIT or REVOKE access to CACCFDSN resources

The administrator can use any of the following methods to control access to CACCFDSN resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: CACCFDSN resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) CACCFDSN(ABCDEFGH)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) CACCFDSN(ABCDEFGH)
```

This example permits users in the Technical Services Department to access ABCDEFGH.XXXXXX on Fridays only:

```
TSS PERMIT(TECHPROF) CACCFDSN(ABCDEFGH.XXXXXX) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access (ABCDEFGH.XXXXXX):

```
TSS PERMIT(TECHPROF) CACCFDSN((ABCDEFGH.XXXXXX)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) CACCFDSN(ABCDEFGH.XXXXXX)
```

CACCFMEM Resource Class—Secure CACCFMEM Members

Valid on z/OS.

Use CACCFMEM to secure CACCFMEM Members.

When used with TSS ADDTO/REMOVE, this resource class has the following format:.

```
TSS ADDTO(acid) CACCFMEM(MEMBER1)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) CACCFMEM(MEMBER1.ABCDEFGH)
                 ACCESS(ALL)
```

Prefix length

One to forty-four characters

CACCFMEM is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CACCFMEM (OWN) authority to ADD or REMOVE ownership of CACCFMEM resources from ACIDs
- CACCFMEM(XAUTH) authority to PERMIT or REVOKE access to CACCFMEM resources

The administrator can use any of the following methods to control access to CACCFMEM resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: CACCFMEM resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) CACCFMEM(MEMBER1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) CACCFMEM(MEMBER1)
```

This example permits users in the Technical Services Department to access MEMBER1.ABCDEFGH on Fridays only:

```
TSS PERMIT(TECHPROF) CACCFMEM(MEMBER1.ABCDEFGH) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access XDT98000:

```
TSS PERMIT(TECHPROF) CACCFMEM(MEMBER1.ABCDEFGH) ACCESS(ALL)
```

This example revokes access :

```
TSS REVOKE(TECHPROF) CACCFMEM(MEMBER1.ABCDEFGH)
```

CACMD Resource Class—Secure CA Commands

Valid on z/OS and z/VM.

Use CACMD to secure other CA Product commands and programs.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) CACMD(`*`prefix(es)`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) CACMD(`*`prefix(es)`*`)`

Prefix length

One to forty-four characters

Capacity of list

One to five prefixes per TSS command

CACMD is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CACMD(OWN) authority to ADD or REMOVE Authority to control access of CA product specific commands owned
- CACMD(XAUTH) authority to PERMIT or REVOKE the Authority to CA product specific commands, functions and programs

The administrator can use any of the following methods to control access to CACMD resources: Expiration, Facility, Time/Day, and Actions.

Examples: CACMD resource class

This example gives PRDCNTL ownership of the BrightStor™ CA-1® command named LOINQUIR:

```
TSS ADDTO(PRDCNTL) CACMD(LOINQUIR)
```

This example removes ownership:

```
TSS REMOVE(PRDCNTL) CACMD(LOINQUIR)
```

This example permit ACID, TAPELIB, the ability to use the BrightStor CA-1 command, CATALOG:

```
TSS PERMIT(TAPELIB) CACMD(CATALOG)
```

This example enables all users to issue the BrightStor CA-1 command INQUIRE:

```
TSS PERMIT(ALL) CACMD(INQUIRE)
```

CAGSVX Resource Class—Secure CAGSVX for SYSVIEW

Valid on z/OS.

Use CAGSVX to secure CAGSVX for SYSVIEW 7.2

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) CAGSVX(ABCDEFGH)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) CAGSVX(ABCDEFGH.XXXXXX)
```

Prefix length

One to forty-four characters

Capacity of list

One to five prefixes per TSS command

CAGSVX is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CAGSVX (OWN) authority to ADD or REMOVE ownership of CAGSVX resources from ACIDs
- CAGSVX(XAUTH) authority to PERMIT or REVOKE access to CAGSVX resources

The administrator can use any of the following methods to control access to CAGSVX resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: CAGSVX resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) CAGSVX(ABCDEFGH)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) CAGSVX(ABCDEFGH)
```

This example permits users in the Technical Services Department to access ABCDEFGH.XXXXXX on Fridays only:

```
TSS PERMIT(TECHPROF) CAGSVX(ABCDEFGH.XXXXXX) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access (ABCDEFGH.XXXXXX):

```
TSS PERMIT(TECHPROF) CAGSVX(ABCDEFGH.XXXXXX)
```

This example revokes access :

```
TSS REVOKE(TECHPROF) CAGSVX(ABCDEFGH.XXXXXX)
```

CALIBMEM Resource Class—Secure CALIBMEM Members

Valid on z/OS and z/VSE.

Use CALIBMEM to secure CALIBMEM members in LIBRARIAN.

When used with TSS ADDTO/REMOVE, this resource class has the following format

`TSS ADDTO(`*`acid`*`) CALIBMEM(`*`member`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) CALIBMEM(MEMBER1.ABCDEFGH)`

Prefix length

One to forty-four characters

Capacity of list

One to five prefixes per TSS command.

CALIBMEM is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CALIBMEM (OWN) authority to ADD or REMOVE ownership of CALIBMEM resources from ACIDs
- CALIBMEM(XAUTH) authority to PERMIT or REVOKE access to CALIBMEM resources

The administrator can use any of the following methods to control access to CALIBMEM resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Example: CALIBMEM resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) CALIBMEM(MEMBER)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) CALIBMEM(MEMBER)
```

This example permits users in the Technical Services Department to access MEMBER on Fridays only:

```
TSS PERMIT(TECHPROF) CALIBMEM(MEMBER) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access MEMBER.ABCDEFGH with access(UPDATE):

```
TSS PERMIT(TECHPROF) CALIBMEM(MEMBER.ABCDEFGH) ACCESS(UPDATE)
```

This example revokes access :

```
TSS REVOKE(TECHPROF) CALIBMEM(MEMBER.ABCDEFGH)
```

CAREPORT Resource Class—Secure Reports

Valid on z/OS.

Use CAREPORT to secure reports for CA-DISPATCH™.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*acid*`) CAREPORT(`*report*`,...)`

Command length

forty-four character report name

Capacity of list

One to five commands per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`) CAREPORT(`*report*`,...)`

Command length

Forty-four characters

Capacity of list

One to five commands per TSS command

CAREPORT is used with:

- The commands ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CAREPORT(OWN) authority to ADD or REMOVE ownership of CAREPORT resources from ACIDs
- CAREPORT(XAUTH) authority to PERMIT or REVOKE access to CAREPORT resources

The administrator can specify any of the following access levels: None, Update, Read, Write, Create, Scratch, and All.

The administrator can use any of the following methods to control access to CAREPORT resources: Expiration, Program Pathing, Time/Day, and Actions.

Masking

Prefixing is supported. Masking is not.

Example: CAREPORT resource class

This example defines the REP99 report to DEPT01:

```
TSS ADDTO(DEPT01) CAREPORT(REP99)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(DEPT01) CAREPORT(REP99)
```

This example grants USER42 the ability to use report REP99:

```
TSS PERMIT(USER42) CAREPORT(REP99)
```

CASECAUT Resource Class—Authorize Restricted Administrative Priviliges

Valid on z/OS.

Use CASECAUT to authorize users with no administrative authorities to change certain password related fields for other users within their scope and issue digital certificate and keyring related commands.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) CASECAUT(prefix,prefix...)`

Prefix length

Two to twenty-six characters

Capacity of list

One to eight prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) CASECAUT(`*`prefix(es)`*`)`

Prefix length

Two to forty-four characters

Capacity of list

One to eight prefixes per TSS command

This keyword uses:

- The commands ADDTO, REMOVE, PERMIT, REVOKE, WHOHAS, and WHOOWNS
- The ACID types User, Profile, Department, DivisionZone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CASECAUT(OWN) authority to ADD or REMOVE ownership of CASECAUT resources
- CASECAUT(XAUTH) authority to PERMIT or REVOKE access to CASECAUT resources

The administrator can:

- Specify the access levels: NONE, USE, UPDATE, PRIVILEG, CONTROL, and GRANT. If ACCESS is not specified, CA Top Secret defaults to USE access
- Use any of the following methods to control access to CASECAUT: Expiration, Facility, Time/Day, and Actions.

Masking

The CASECAUT resource class supports all masking characters.

Examples: CASECAUT resource class

- This example adds ownership of prefix TSSCMD to SYSDEPT acid:

  ```
  TSS ADDTO(SYSDEPT) CASECAUT(TSSCMD)
  ```

- This example removes ownership:

  ```
  TSS REMOVE(SYSDEPT) CASECAUT(TSSCMD)
  ```

- This example permits USER01 to change the password field for other users within scope:

  ```
  TSS PERMIT(USER01) CASECAUT(TSSCMD.USER.REPLACE.PASSWORD)
  ```

- This example revokes access:

  ```
  TSS REVOKE(USER01) CASECAUT(TSSCMD.USER.REPLACE.PASSWORD)
  ```

CATAPE Resource Class—Secure Tape Volumes

Valid on z/OS and z/VM.

Use CATAPE to secure access to tape volumes for BrightStor CA-1.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) CATAPE(`*`prefix(es)`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) CATAPE(`*`prefix(es)`*`)`

Prefix length

One to forty-four characters

Capacity of list

One to five prefixes per TSS command

CATAPE is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CATAPE(OWN) authority to ADD or REMOVE Authority to control access of CA product specific commands owned
- CATAPE(XAUTH) authority to PERMIT or REVOKE the Authority to CA product specific commands, functions and programs

The administrator can:

- Specify the accesses ALL, UPDATE, READ, WRITE, and NONE.
- Use any of the following methods to control access to CATAPE resources: Expiration, Facility, Time/Day, and Actions.

Example: CATAPE resource class

This example gives PRDCNTL ownership of the CATAPE named BLPNORES:

```
TSS ADDTO(PRDCNTL) CATAPE(BLPNORES)
```

This example removes ownership:

```
TSS REMOVE(PRDCNTL) CATAPE(BLPNORES)
```

This example gives SYSPRG ownership of the BrightStor™ CA-DYNAM®/B command named BACKUP:

```
TSS ADDTO(SYSPRG) CATAPE(DYNAMB.OPER.CMD.BACKUP)
```

This example removes ownership:

```
TSS REMOVE(SYSPRG) CATAPE(DYNAMB.OPER.CMD.BACKUP)
```

This example permits ACID, TAPELIB, the ability to use the CATAPE resource, DEACT:

```
TSS PERMIT(TAPELIB) CATAPE(DEACT)
                    ACCESS(READ)
```

This example enables all users to use the CATAPE resource REINIT:

```
TSS PERMIT(ALL) CATAPE(REINIT)
                ACCESS(READ)
```

This example permits ACID, VMUSER1, the ability to use the BrightStor™ CA-DYNAM®/B command, BACKUP:

```
TSS PERMIT(VMUSER1) CATAPE(DYNAMB.OPER.CMD.BACKUP)
```

This example enables all users to issue the BrightStor™ CA-DYNAM®/B command RESTORE:

```
TSS PERMIT(ALL) CATAPE(DYNAMB.USER.CMD.RESTORE)
```

CAVAPPL Resource Class—Secure CAVAPPL for CAVMAN

Valid on z/OS.

Use CAVAPPL to secure CAVAPPL for CAVMAN.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) CAVAPPL(APPLABCD)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) CAVAPPL(APPLABCD.XXXXXXXX)
```

Prefix length

One to forty-four characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CAVAPPL (OWN) authority to ADD or REMOVE ownership of CAVAPPL resources from ACIDs
- CAVAPPL(XAUTH) authority to PERMIT or REVOKE access to CAVAPPL resources

The administrator can use any of the following methods to control access to CAVAPPL resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: CAVAPPL resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) CAVAPPL(APPLABCD)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) CAVAPPL(APPLABCD)
```

This example permits users in the Technical Services Department to access APPLABCD.XXXXXXXX on Fridays only:

```
TSS PERMIT(TECHPROF) CAVAPPL(APPLABCD.XXXXXXXX)
                     DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access APPLABCD.XXXXXXXX.

```
TSS PERMIT(TECHPROF) CAVAPPL(APPLABCD.XXXXXXXX)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) CAVAPPL(APPLABCD.XXXXXXXX)
```

CBIND Resource Class—Secure CBIND for z/OS Objects

Valid on z/OS.

Use CBIND to secure CBIND for z/OS objects.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*acid*`) CBIND(OBJECT1)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`) CBIND(OBJECT1.XXXXXXXX)`

Prefix length

One to sixteen characters

Capacity of list

One to five prefixes per TSS command.

CBIND is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CBIND (OWN) authority to ADD or REMOVE ownership of CBIND resources from ACIDs
- CBIND(XAUTH) authority to PERMIT or REVOKE access to CBIND resources

The administrator can use any of the following methods to control access to CBIND resources: Expiration, Facility, Program Pathing, Time/Day, Actions.

Examples: CBIND resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) CBIND(OBJECT1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) CBIND(OBJECT1)
```

This example permits users in the Technical Services Department to access OBJECT1.XXXXXXXX on Fridays only:

```
TSS PERMIT(acid) CBIND(OBJECT1.XXXXXXXX)
                 DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access OBJECT1.XXXXXXXX

```
TSS PERMIT(acid) CBIND(OBJECT1.XXXXXXXX)
```

This example revokes access :

```
TSS REVOKE(TECHPROF) CBIND(OBJECT1.XXXXXXXX)
```

CIMS Resource Class—Secure IMS Commands

Valid on z/OS.

Use CIMS to secure IMS commands.

This resource class has the following format for TSS ADDTO/REVOKE

```
TSS ADDTO(acid) CIMS(command,...)
```

Command length

Three-character IMS verb

Capacity of list

One to five commands per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) CIMS(command,...)
```

Command length

Three characters

Capacity of list

One to five commands per TSS command

CIMS is used with:

- The commands ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CIMS(OWN) authority to ADD or REMOVE ownership of CIMS resources from ACIDs
- CIMS(XAUTH) authority to PERMIT or REVOKE access to CIMS resources

The administrator can:

- Use any of the following methods to control access to CIMS resources: Expiration, Program Pathing, Time/Day, and Actions
- Specify the access levels None and ALL

Prefixing is supported. Masking is not.

Examples: CIMS resource class

This example defines the IMS/DISPLAY command to DEPT01:

```
TSS ADDTO(DEPT01) CIMS(DIS)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(DEPT01) CIMS(DIS)
```

This example grants USER42 the ability to use the IMS/START command:

```
TSS PERMIT(USER42) CIMS(STA)
```

CPCMD Resource Class—Secure CP Commands

Valid on z/VM.

Use CPCMD to secures CP commands issued on z/ VM.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) CPCMD(command,command,...)
```

Prefix length

One to eight characters

Capacity of list

One to five CP command per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) CPCMD(command,command,...)
```

Prefix length

One to eight characters

Capacity of list

One to five CP commands per TSS command

CPCMD is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CPCMD(OWN) authority to ADDTO or REMOVE ownership of CPCMDs from ACIDs
- CPCMD(XAUTH) authority to PERMIT or REVOKE access to CPCMDs

The administrator can use any of the following methods to control access to CPCMDs: Expiration, Facility, Time/Day, and Actions. The CPCMD resource also supports use of the ACTION(VMPRIV) attribute.

Examples: CPCMD resource class

This example protects the CP command, SHUTDOWN, by assigning ownership to the Operations Department:

```
TSS ADDTO(OPERDEPT) CPCMD(SHUTDOWN)
```

The administrator may now PERMIT limited access to users or to profiles that require access.

This example removes ownership of CP commands:

```
TSS REMOVE(OPERDEPT) CPCMD(SHUTDOWN)
```

This example permits USER03 to use the CP command QUERY, to get a listing of real DASD on the real computer:

```
TSS PERMIT(USER03) CPCMD(QUERY.DASD)
                   ACTION(VMPRIV)
```

If ACTION(VMPRIV) was not specified, USER03 would have gotten a listing of virtual DASD on his virtual machine.

This example permits USER03 to use the CP command, SHUTDOWN:

```
TSS PERMIT(USER03) CPCMD(SHUTDOWN)
```

In this example, VMPRIV is implied.

This example revokes USER03's permission to use SHUTDOWN:

```
TSS REVOKE(USER03) CPCMD(SHUTDOWN)
```

CPU Resource Class—Secure CPU Access

Valid on z/OS and z/VM.

Use CPU to:

- Secure an ACID's access to a particular CPU
- Force an ACID to enter the system only through a particular CPU

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) CPU(smfid|id from DMKSYS)
```

Prefix length

One to four characters for z/OS, One to eight characters for VM

Capacity of list

One to five CPU IDs per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) CPU(smfid)
```

Prefix length

One to four characters for z/OS, One to eight characters for VM

Capacity of list

One to five CPU IDs per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CPU(OWN) authority to ADD or REMOVE ownership of CPU from ACIDs
- CPU(XAUTH) authority to PERMIT or REVOKE access to CPUs

The administrator can use any of the following methods to control access to CPUs: Expiration, Facility, Program Pathing, Time/Day, and Actions. The administrator can use any of the following methods to control access to CPUs details on each of the access control methods.

Examples: CPU resource class

This example protects the CPU, SYSA, byr assigning ownership to Department 01:

```
TSS ADDTO(DEPT01) CPU(SYSA)
```

The administrator may now PERMIT access to users or to profiles that require access.

This example removes ownership:

```
TSS REMOVE(DEPT01) CPU(SYSA)
```

This example permits the day shift to execute BATCH jobs on CPU-SYSA:

```
TSS PERMIT(DAYPROF) CPU(SYSA) FACILITY(BATCH)
```

This example revokes access to SYSA:

```
TSS REVOKE(DAYPROF) CPU(SYSA)
```

CSFKEYS Resource Class—Secure ICSF CSFKEYS

Valid on z/OS.

Use CSFKEYS to secure CSFKEYS for z/OS Integrated Cryptographic Service Facility (ICSF). For information on setting the keys, see IBM's *OS/390 Integrated Cryptographic Service Facility: Administration Guide*

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) CSFKEYS(CSFKEYS1)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) CSFKEYS(CSFKEYS1)
```

Prefix length

One to eight characters

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS.
- The ACID types User, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CSFKEYS (OWN) authority to ADD or REMOVE ownership of CSFKEYS resources from ACIDs
- CSFKEYS(XAUTH) authority to PERMIT or REVOKE access to CSFKEYS resources

The administrator can use any of the following methods to control access to CSFKEYS resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: CSFKEYS resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) CSFKEYS(CSFKEYS1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) CSFKEYS(CSFKEYS1)
```

This example permits users to access CSFKEYS(CSFKEYS1) on Fridays only:

```
TSS PERMIT(TECHUSER) CSFKEYS(CSFKEYS1)
                     DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access CSFKEYS1:

```
TSS PERMIT(TECHPROF) CSFKEYS(CSFKEYS1)
```

This example revokes access :

```
TSS REVOKE(TECHUSER) CSFKEYS(CSFKEYS1)
```

CSFSERV Resource Class—Secure ICSF CSFSERV

Valid on z/OS.

Use CSFSERV to secure CSFSERV for z/OS Integrated Cryptographic Service Facility (ICSF). For information on setting the keys, see IBM's *OS/390 Integrated Cryptographic Service Facility: Administration Guide*.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) CSFSERV(CSFSERV1)
```

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) CSFSERV(CSFSERV1)
```

Prefix length

One to eight characters

CSFSERV is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CSFSERV (OWN) authority to ADD or REMOVE ownership of CSFSERV resources from ACIDs
- CSFSERV(XAUTH) authority to PERMIT or REVOKE access to CSFSERV resources

The administrator can use any of the following methods to control access to CSFSERV resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: CSFSERV resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) CSFSERV(CSFSERV1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) CSFSERV(CSFSERV1)
```

This example permits users to access CSFSERV(CSFSERV1) on Fridays only:

```
TSS PERMIT(TECHUSER) CSFSERV(CSFSERV1)
                     DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access CSFSERVR1:

```
TSS PERMIT(TECHPROF) CSFSERV(CSFSERV1)
```

This example revokes access:

```
TSS REVOKE(TECHUSER) CSFSERV(CSFSERV1)
```

CTSDB2 Resource Class—Secure CTS DB2ENTRY and DB2TRAN Resources

Valid on z/OS.

Use CTSDB2 to secure DB2ENTRY and DB2TRAN resources in CICS Transaction Server (CTS) 1.2 and above.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) CTSDB2(prefix(es))
```

Prefix length

One to eight characters

Capacity of list

One to five CICS DB2 resource prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) CTSDB2(prefix(es))
                 ACCESS(access levels)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

CTSDB2 is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- CTSDB2(OWN) authority to ADD or REMOVE ownership of CTSDB2s from ACIDs
- CTSDB2(XAUTH) authority to PERMIT or REVOKE access to CICS DB2 resources

The administrator can:

- Specify any or all of the following access levels: SET, INQUIRE, ALL, BROWSE, DELETE, NONE, READ, UPDATE, WRITE, INSTALL, and COLLECT. If ACCESS is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to CICS DB2 resources: Expiration, Facility, PPT Pathing, Time/Day, and Actions.

DATABASE Resource Class—Secure Databases

Valid on z/OS.

Use DATABASE to secure all databases.

This resource class has the format with TSS ADDTO/REMOVE:

```
TSS ADDTO(acid) DATABASE(DdddFfff)
```

The letters ddd represent the database number from 1 to 255 and fff is the file number from 1 to 255.

Prefix length

One to eight characters

Capacity of list

One to five ADABAS prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) DATABASE(DdddFfff)
                 ACCESS(access levels)
```

Prefix length

One to eight characters

Capacity of list

One to five ADABAS prefixes per TSS command

DATABASE is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS.
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, LSCA, ZCA, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DATABASE(OWN) authority to ADD or REMOVE ownership of DATABASE from ACIDs
- DATABASE(XAUTH) authority to PERMIT or REVOKE access to DATABASEs

The administrator can:

- Specify any or all of the following access levels for ADABAS databases and files: ALL, CREATE, NONE, READ, SCRATCH, UPDATE, and WRITE. If ACCESS is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to databases: Expiration, Facility, Time/Day, and Actions.

Examples: DATABASE resource class

This example protects the file number 105 contained within the ADABAS database number 3, by assigning ownership to DEPT01:

```
TSS ADDTO(DEPT01) DATABASE(D003F105)
```

This example remove ownership:

```
TSS REMOVE(DEPT01) DATABASE(D003F105)
```

This example permits USER01 to file number 105 contained within ADABAS database number 3:

```
TSS PERMIT(USER01) DATABASE(D003F105)
```

This example revokes access:

```
TSS REVOKE(USER01) DATABASE(D003F105)
```

DB2 Resource Class—Secure DB2 Databases

Valid on z/OS.

Use DB2 to secure a DB2 database.

For connection authorization, individual ACID checking is not performed for CICS and IMS but instead for the ACID associated with the CICS and IMS started tasks or the associated MASTFAC ACID. The ACID associated with a job is used to validate the connection for TSO and BATCH.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) DB2(DSNR.)
TSS ADDTO(acid) DB2(DXT.dddd)
```

DSNR

Class name provided for a DB2 resource for DB2 and subsystem connection.

DXT

Access to data elements through the Data Extract Utility.

dddd

One to four character data element (file, PCB, view)

Prefix length

One to eight characters

Capacity of list

One to five DB2 resources per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) DB2(DSNR.ssss.BATCH)
TSS PERMIT(acid) DB2(DSNR.ssss.DIST)
TSS PERMIT(acid) DB2(DSNR.ssss.MASS)
TSS PERMIT(acid) DB2(DSNR.ssss.SASS)
TSS PERMIT(acid) DB2(DXT.dddd)
```

DSNR

Class name provided for a DB2 resource for DB2 and subsystem connection.

ssss

The site's subsystem name for DB2.

BATCH

For BATCH and TSO connections.

DIST

For Distributed Data Facility (DDF).

MASS

For IMS connection.

SASS

For CICS connection.

DXT

For access to data elements through the Data Extract Utility.

dddd

One to four character data element (file, PCB, view).

Prefix length

One to forty-four characters per prefix

Capacity of list

One to five prefixes per TSS command

DB2 is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DB2 ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DB2(OWN) authority to ADDTO or REMOVE ownership of DB2 resources from ACIDs
- DB2(XAUTH) authority to PERMIT or REVOKE access to DB2 resources

The administrator can use any of the following methods to control access to DB2 resources: Expiration, Facility, Time/Day, and Actions.

Example: DB2 resource class

This example protects access to an IMS system with the IMS ID of IMSB, by assigning ownership to the Software Department:

```
TSS ADDTO(SOFTDEP) DB2(IMSB.MASS)
```

The administrator may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(SOFTDEP) DB2(IMSB.MASS)
```

This example permits a user (GKM75) to access an IMS system with the IMSID of IMSB:

```
TSS PERMIT(GKM75) DB2(IMSB.MASS)
```

This example revokes access:

```
TSS REVOKE(GKM75) DB2(IMSB.MASS)
```

DB2BUFFP Resource Class—Secure DB2 Buffer Pools

Valid on z/OS.

Use DB2BUFFP to secure DB2 buffer pools.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(acid) DB2BUFFP(buffer pool,buffer pool,buffer pool,...)`

Prefix length

One to five characters

Capacity of list

One to five DB2 buffer pools or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(acid) DB2BUFFP(buffer pool prefix(es))`

Prefix length

One to five characters per prefix

Full name

One to five characters

Capacity of list

One to five buffer pools or prefixes per TSS command

DB2BUFFP is used with:

- The commands CREATE, ADDTO, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DB2BUFFP ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DB2BUFFP(OWN) Authority

The administrator can:

- Specify the accesses: ALL, USE, and NONE. The default is USE.
- Use any of the following methods to control access to buffer pools: Expiration, Facility, Time/Day, and Actions.

Examples: DB2BUFFP resource class

This example gives the Operations Department (OPRDEPT) ownership of the buffer pool BP1:

```
TSS ADDTO(OPRDEPT) DB2BUFFP(BP1)
```

This example removes ownership of the table BP1:

```
TSS REMOVE(OPRDEPT) DB2BUFFP(BP1)
```

This example authorizes USRMARK to access buffer pool BP1 with USE access so that this user can name BP1 in a CREATE INDEX or CREATE TABLESPACE statement:

```
TSS PERMIT(USRMARK) DB2BUFFP(BP1)
                    ACCESS(USE)
```

This example revokes USRMARK's Authority to buffer pool BP1:

```
TSS REVOKE(USRMARK) DB2BUFFP(BP1)
```

DB2COLL Resource Class—Secure DB2 Collection ID

Valid on z/OS.

Use DB2COLL to restrict who can add packages to a particular DB2 collection ID.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DB2COLL(`*`collection-id`*`)`

Prefix length for collection-id

Two to 18 characters

Capacity of list

One to five DB2 collections per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DB2COLL(`*`collection-id`*`)`

Prefix length for collection-id

Two to 18 characters

Capacity of list

One to five collections per TSS command

This resource class has the following format for TSS ADMIN/DEADMIN:

`TSS ADMIN(`*`acid`*`) DB2COLL(`*`Authority level(s)`*`)`

DB2COLL is used with:

- The commands CREATE, DELETE, ADDTO, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, WHOHAS, and REMOVE
- The DB2COLL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA.
- DB2COLL(OWN) Authority

Administrators can:

- Specify the authority levels: OWN, XAUTH, AUDIT, INFOR, REPORT, and ALL.
- Specify the accesses: CREATEIN, PACKADM, ALL, and NONE. The default is NONE.
- Use the Expiration, Facility, Time/Day, and Actions to control access to collections

Examples: DB2COLL resource class

This example gives the Operations Department (OPRDEPT) ownership of collection FACCT:

```
TSS ADDTO(OPRDEPT) DB2COLL(FACCT)
```

This example removes ownership of the collection:

```
TSS REMOVE(OPRDEPT) DB2COLL(FACCT)
```

This example authorizes USRMARK the privilege to include new packages within the FACCT collection:

```
TSS PERMIT(USRMARK) DB2COLL(FACCT) ACCESS(CREATIN,PACKADM)
```

This example revokes USRMARK's Authority:

```
TSS REVOKE(USRMARK) DB2COLL(FACCT) ACCESS(CREATIN,PACKADM)
```

This example gives FINVCA the ability to assign ownership of collections, to permit users to access collections with his scope, and to audit the use of collections owned by his division:

```
TSS ADMIN(FINVCA) DB2COLL(OWN,XAUTH,INFO)
```

This example removes FINVCA's Authority for collections:

```
TSS DEADMIN(FINVCA) DB2COLL(OWN,XAUTH,INFO)
```

This example determines who has access to the FACCT collection:

```
TSS WHOHAS DB2COLL(FACCT)
```

CA Top Secret responds by displaying all of the ACIDs that have access to this particular collection.

This example determines who owns the collection:

```
TSS WHOOWNS DB2COLL(COLLECTION-ID)
```

DB2DBASE Resource Class—Secure DB2 Databases

Valid on z/OS.

Use DB2DBASE to secure DB2 databases.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DB2DBASE(`*`database`*`,...)`

Prefix length

Two to eight characters

Capacity of list

One to five DB2 database or prefixes per TSS command

This resource class has the following format for TSS PERMIT REVOKE:

`TSS PERMIT(`*`acid`*`) DB2DBASE(`*`database prefix(es)`*`)`

Prefix length

Two to eight characters per prefix

Full name

Two to eight characters

Capacity of list

One to five databases or prefixes per TSS command

DB2DBASE is used with:

- The commands CREATE, REMOVE, ADDTO, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DB2DBASE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- DB2DBASE(OWN) Authority

The administrator can:

- Use any of the following methods to control access to databases: Expiration, Facility, Time/Day, and Actions.
- Specify the accesses: DBADM, DBCTRL, DBMAINT, CRETAB, CRETS, DISPDB, DROP, IMAGCOPY, LOAD, RECOVDB, REORG, REPAIR, STARTDB, STATS, STOPDB, ALL, and NONE.

The DB2DBASE resource class supports all masking characters.

Examples: DB2DBASE resource class

This example gives the Finance Department (FINDEPT) ownership of the database ACCTING:

```
TSS ADDTO(FINDEPT) DB2DBASE(ACCTING)
```

This example removes ownership of the database ACCTING :

```
TSS REMOVE(FINDEPT) DB2DBASE(ACCTING)
```

This example authorizes USRJANE to create new tables on database ACCTING:

```
TSS PERMIT(USRJANE) DB2DBASE(ACCTING)
                    ACCESS(CRETAB)
```

This example revokes USRJANE's Authority to access databases ACCTING:

```
TSS REVOKE(USRJANE) DB2DBASE(ACCTING)
```

DB2PKG Resource Class—Secure DB2 Packages

Valid on z/OS.

Use DB2PKG to secure DB2 packages.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DB2PKG(`*`collection-id.package-id`*`,...)`

Prefix length

Two to twenty-six characters

Capacity of list

One to five DB2 packages per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DB2PKG(`*`collection-id.package-id`*`,...)`

Prefix length

Two to twenty-six characters

Full name length

Two to 27 characters

Capacity of list

One to five packages per TSS command

This resource class has the following format for TSS ADMIN/DEADMIN:

`TSS ADMIN(`*`acid`*`) DB2PKG(`*`Authority level(s)`*`)`

DB2PKG is used with:

- The commands CREATE, DELETE, ADDTO, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, WHOHAS, and REMOVE
- The DB2PKG ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- DB2PKG(OWN) Authority

The administrator can specify:

- The accesses: BIND, COPY, EXECUTE, ALL, and NONE. The default is EXECUTE
- The following methods to control access to packages: Expiration, Facility, Time/Day, and Actions
- Authority levels: OWN, XAUTH, AUDIT, INFO, REPORT, and ALL

Examples: DB2PKG resource class

This example gives the Operations Department (OPRDEPT) ownership of package PACK2, in the FIN collection:

```
TSS ADDTO(OPRDEPT) DB2PKG(FIN.PACK2)
```

This example removes ownership of this package:

```
TSS REMOVE(OPRDEPT) DB2BUFFP(FIN.PACK2)
```

This example authorizes USRMARK the Authority to copy the contents of a specified package:

```
TSS PERMIT(USRMARK) DB2PKG(FIN.PACK1) ACCESS(COPY)
```

This example revokes USRMARK's Authority:

```
TSS REVOKE(USRMARK) DB2PKG(FIN.PACK1) ACCESS(COPY)
```

This example gives OPRDEPT the ability to assign ownership of packages, to permit users to access packages with his scope, and audit the use of packages owned by the division:

```
TSS ADMIN(OPRDEPT) DB2PKG(OWN,XAUTH,INFO)
```

This example removes OPRDEPT's Authority for packages:

```
TSS DEADMIN(OPRDEPT) DB2PKG(OWN,XAUTH,INFO)
```

This example determines who has access to the PACK3 package in the FIN collection:

```
TSS WHOHAS DB2PKG(FIN.PACK3)
```

CA Top Secret responds by displaying all of the ACIDs that have access to this particular package.

This example determines who owns package PACK3 in the FIN collection:

```
TSS WHOOWNS DB2PKG(FIN.PACK3)
```

DB2PLAN Resource Class—Secure DB2 Plans

Valid on z/OS.

Use DB2PLAN to secure DB2 plans.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DB2PLAN(`*`plan`*`,`*`plan`*`,`*`plan`*`,...)`

Prefix length

One to eight characters

Capacity of list

One to five DB2 plans or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DB2PLAN(`*`plan prefix(es)`*`)`

Prefix length

One to eight characters per prefix

Capacity of list

One to five plans or prefixes per TSS command

DB2PLAN is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, WHOHAS, and DELETE.
- The DB2PLAN ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA.
- DB2PLAN(OWN) Authority

The administrator can specify:

- The accesses: BIND, EXECUTE, ALL, and NONE. The default is EXECUTE.
- The following methods to control access to plans: Expiration, Facility, Time/Day, and Actions.

Examples: DB2PLAN resource class

This example gives the Engineering Department (ENGDEPT) ownership of the plan SR19052P:

```
TSS ADDTO(ENGDEPT) DB2PLAN(SR19052P)
```

This example removes ownership of the plan SR19052P:

```
TSS REMOVE(ENGDEPT) DB2PLAN(SR19052P)
```

This example authorizes USRMIKE to bind on plan SR19052P:

```
TSS PERMIT(USRMIKE) DB2PLAN(SR15092P) ACCESS(BIND)
```

This example revokes USRMIKE's Authority to plan SR19052P:

```
TSS REVOKE(USRMIKE) DB2PLAN(SR19052P)
```

DB2SEQ Resource Class—Identify DB2 Sequences

Valid on z/OS.

Use DB2SEQ to identify DB2 sequences.

This resource class has the following format for TSS ADDTO or REMOVE.

```
TSS ADD(acid) DB2SEQ(schema.seq-id,schema.seq-id,...)
```

Prefix length

2-26 characters

Capacity of list

1-5 DB2 sequence names per TSS command.

This resource class has the following format:

```
TSS PER(acid) DB2SEQ(schema.seq-id,schema.seq-id,...)
```

Prefix length

2-26 characters

Full name length

2-255 characters

Capacity of list

1-5 sequence names per TSS command.

This resource class has the following format for TSS ADMIN or DEADMIN:

```
TSS ADMIN(acid) DB2SEQ(authority level(s))
```

Administrators can specify the authority levels: OWN, XAUTH, AUDIT, INFO, REPORT, ALL.

DB2SEQ is used with:

- The commands CREATE, DELETE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DB2SEQ ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DB2SEQ ACID types User, DCA, VCA, LSCA, ZCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DB2SEQ(OWN) authority

The DB2SEQ resource class supports all masking characters.

The administrator can use any of the following methods to control access to DB2 sequences: Expiration, Facility, Time/Day, and Actions.

DB2STOGP Resource Class—Secure DB2 Storage Groups

Valid on z/OS.

Use DB2STOGP to secure DB2 storage groups.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DB2STOGP(`*`storage group, storage group,storage group,...`*`)`

Prefix length

One to eight characters

Capacity of list

One to five DB2 storage groups or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DB2STOGP(`*`storage group prefix(es)`*`)`

Prefix length

One to eight characters per prefix

Full name

One to eight characters

Capacity of list

One to five storage groups or prefixes per TSS command

DB2STOGP is used with:

- The commands CREATE, ADDTO, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DB2STOGP ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DB2STOGP ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DB2STOGP(OWN) Authority

The administrator can:

- Specify the accesses ALL and USE.
- The following methods to control access to storage groups: Expiration, Facility, Time/Day, and Actions.

Examples: DB2STOGP Resource Class

This example gives the Operations Department (OPRDEPT) ownership of the storage group TESTDASD:

```
TSS ADDTO(OPRDEPT) DB2STOGP(TESTDASD)
```

This example removes ownership of the table TESTDASD:

```
TSS REMOVE(OPRDEPT) DB2STOGP(TESTDASD)
```

This example authorizes USRMARK to access storage group TESTDASD:

```
TSS PERMIT(USRMARK) DB2STOGP(TESTDASD)
```

This example revokes USRMARK's Authority to buffer pool TESTDASD:

```
TSS REVOKE(USRMARK) DB2STOGP(TESTDASD)
```

DB2SYS Resource Class—Secure DB2 Privileges and Authorities

Valid on z/OS.

Use DB2SYS to secure DB2 system privileges and authorities.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DB2SYS(`*`priv`*`,`*`priv`*`,`*`priv`*`,...)`

Prefix length

One to eight characters

Capacity of list

One to five DB2 system privileges or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DB2SYS(`*`priv`*`|BINDAGENT.`*`owner-id`*`,...)`

Prefix length

One to eight characters per prefix

Full name

1-18 characters

Capacity of list

One to five system privileges or prefixes per TSS command

DB2SYS is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DB2SYS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DB2SYS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DB2SYS(OWN) Authority

The administrator can use any of the following methods to control access to system privileges: Expiration, Facility, Time/Day, and Actions.

Privileges

The administrator can specify the privileges: SYSADM, SYSOPR, BINDADD, BSDS, CREDBC, CRESG, DISPLAY, MONITOR1, MONITOR2, RECOVER, STOPALL, STOSPACE, TRACE, CREALIAS, SYSCTRL, ARCHIVE, and BINDAGENT.

Note: Unlike other DB2SYS privileges that have global scope, the BINDAGENT privilege only grants the holder the bind agent Authority for a specific .

Examples: DB2SYS resource class

This example gives the Investment Department (INVDEPT) ownership of the SYSADM privilege:

```
TSS ADDTO(INVDEPT) DB2SYS(SYSADM)
```

This example removes ownership of a system privilege:

```
TSS REMOVE(INVDEPT) DB2SYS(SYSADM)
```

This example authorizes USRJIM to create a storage group:

```
TSS PERMIT(USRJIM) DB2SYS(CRESG)
```

This example revokes USRJIM's Authority to create a storage group:

```
TSS REVOKE(USRJIM) DB2SYS(CRESG)
```

This example authorizes USRMARK as a bind agent for USRMIKE's packages:

```
TSS PERMIT(USRMARK) DB2SYS(BINDAGENT.USRMIKE)
```

This example revokes USRMARK's Authority as USRMIKE's bind agent:

```
TSS REVOKE(USRMARK) DB2SYS(BINDAGENT.USRMIKE)
```

DB2TABLE Resource Class—Secure DB2 Tables

Valid on z/OS.

Use DB2TABLE to secure DB2 tables.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DB2TABLE(`*`userid.table/view`*`,...)`

Prefix length

two to twenty-six characters

Capacity of list

One to five DB2 tables or views or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DB2TABLE(`*`userid.table/view prefix(es)`*`)`

Prefix length

Two to twenty-six characters per prefix

Full name

Two to forty-six characters

Capacity of list

One to five tables or views or prefixes per TSS command

DB2TABLE is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, DELETE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DB2TABLE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DB2TABLE ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DB2TABLE(OWN) Authority

The administrator can:

- Specify the accesses: ALL, NONE, ALTER, CREATE, DELETE, INDEX, INSERT, SELECT, and UPDATE. CREATE access only applies to Views.
- Use the following methods to control access to tables or views: Expiration, Facility, Time/Day, and Actions.

The DB2TABLE resource class supports all masking characters.

Examples: DB2TABLE Resource Class

This example gives the Payroll Department (PAYDEPT) ownership of the table USRMIKE.PAYR:

```
TSS ADDTO(PAYDEPT) DB2TABLE(USRMIKE.PAYR)
```

This example removes ownership of the table PAYR:

```
TSS REMOVE(PAYDEPT) DB2TABLE(USRMIKE.PAYR)
```

This example authorizes USRMARE to select the social security information on table USRMIKE.PAYR:

```
TSS PERMIT(USRMARE) DB2TABLE(USRMIKE.PAYR) ACCESS(SELECT)
```

This example revokes USRMARE's Authority to table USRMIKE.PAYR:

```
TSS REVOKE(USRMARE) DB2TABLE(USRMIKE.PAYR)
```

This example restricts a user's ability to update specifically-named columns in a table. Authorize USRGAB with update privilege to correct any errors in the start-date column on the USRMIKE.PAYR table:

```
TSS PERMIT(USRGAB) DB2TABLE(USRMIKE.PAYR.STRTDTE) ACCESS(UPDATE)
```

This example revokes USRGAB's Authority to table USRMIKE.PAYR:

```
TSS REVOKE(USRGAB) DB2TABLE(USRMIKE.PAYR)
```

DB2TABSP Resource Class—Secure DB2 Table Spaces

Valid on z/OS.

Use DB2TABSP to secure DB2 table spaces.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DB2TABSP(`*`database.table space`*`,...)`

Prefix length

Two to 17 characters

Capacity of list

One to five DB2 table spaces or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DB2TABSP(`*`database.table space prefix(es)`*`)`

Prefix length

Two to 17 characters per prefix

Full name

Two to 17 characters

Capacity of list

One to five table spaces or prefixes per TSS command

The DB2TABSP resource class supports all masking characters.

DB2TABSP is used with:

- The commands CREATE, DELETE, ADDTO, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DB2TABSP ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- The DB2TABSP ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DB2TABSP ACID types User, DCA, VCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DB2TABSP(OWN) Authority

Administrators can specify:

- Any or all of the following levels: OWN, XAUTH, AUDIT, INFO, REPORT, ALL.
- The administrator can specify the accesses: USE, NONE, and ALL. The default is USE.

Examples: DB2TABSP resource class

This example gives the Finance Department (FINDEPT) ownership of the table space TSPACE1 that resides on database ACCTING:

```
TSS ADDTO(FINDEPT) DB2TABSP(ACCTING.TSPACE1)
```

This example removes ownership of the table space TSPACE1:

```
TSS REMOVE(FINDEPT) DB2TABSP(ACCTING.SPACE1)
```

This example authorizes USRMARK to access table space ACCTING.TSPACE1:

```
TSS PERMIT(USRMARK) DB2TABSP(ACCTING.TSPACE1)
```

This example revokes USRMARK's Authority to table space ACCTING.TSPACE1:

```
TSS REVOKE(USRMARK) DB2TABSP(ACCTING.TSPACE1)
```

This example gives FINVCA the ability to assign ownership of table spaces, to PERMIT user to access table spaces within his scope, and to audit the use of table spaces owned by the division:

```
TSS ADMIN(FINVCA) DB2TABSP(OWN,XAUTH,INFO)
```

This example removes FINVCA's Authority for table spaces:

```
TSS DEADMIN(FINVCA) DB2TABSP(OWN,XAUTH,INFO)
```

DBD Resource Class—Secure IMS Database Name

Valid on z/OS.

Use DBD to secure an IMS database descriptor name.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) DBD(prefix(es))
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) DBD(prefix(es)) ACCESS(access levels)
```

Prefix length

One to eight characters per prefix

Capacity of list

One to five prefixes per command

DBD is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DBD ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DBD ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DBD(OWN) authority to ADD or REMOVE ownership of DBD resources from ACIDs
- DBD(XAUTH) authority to PERMIT or REVOKE access to DBD

The administrator can:

- Use following methods to control access to DBDs: Expiration, Facility, PRIVPGM, PSB Pathing, Time/Day, and Actions
- Specify the access levels: INQUIRE, SET, DELETE, REPLACE, NONE, READ, UPDATE, and WRITE. If ACCESS is not specified, CA Top Secret defaults to READ access.

Examples: DBD resource class

This example gives the Personnel Department ownership of the DBD resource for the corporate personnel database (PERSFILE):

```
TSS ADDTO(PERSDPT) DBD(PERSFILE)
```

The administrator may now PERMIT access to users or profiles that require access:

This example removes ownership:

```
TSS REMOVE(PERSDPT) DBD(PERSFILE)
```

This example permits user PERSF1 to update DBDs used by the Personnel Department:

```
TSS PERMIT(PERSF1) DBD(PERS44,PERS51,PERS07) ACCESS(U)
```

This example revokes access:

```
TSS REVOKE(PERSF1) DBD(PERS44,PERS51)
```

Now, user PERSF1 may only update PERS07.

DCSS Resource Class—Secure Segments

Valid on z/VM.

Use DCSS to secure discontiguous Saved Segments/Named Saved Systems (NSS).

Note: DCSS/NSS protection does not apply to DCSSs/NSSs that are IPLed.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) DCSS(name,name,...)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) DCSS(name(s))
                 ACCESS(access levels)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

DCSS is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DCSS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DCSS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DCSS(OWN) authority to ADD or REMOVE ownership of DCSSs from ACIDs
- DCSS(XAUTH) authority to PERMIT or REVOKE access to DCSSs

The administrator can:

- Specify any or all of the following access levels: SHR, NOSHR, FIND, PURGE, ALL, and NONE. If ACCESS is not specified, CA Top Secret defaults to SHR access.
- Use the following methods to control access to DCSSs: Expiration, Facility, Time/Day, and Actions.

Example: DCSS resource class

This example protects the DCSS named SAS by assigning ownership to the Engineering Department (ENGDEPT):

```
TSS ADDTO(ENGDEPT) DCSS(SAS)
```

This example removes ownership:

```
TSS REMOVE(ENGDEPT) DCSS(SAS)
```

This example permits a user in the Engineering Department to use the program product, SAS:

```
TSS PERMIT(ENGUSER) DCSS(SAS)
```

This example revokes access:

```
TSS REVOKE(ENGUSER) DCSS(SAS)
```

DCT Resource Class—Secure CICS DCT Data

Valid on z/OS and z/VSE.

Use DCT to secure transient data entries in the CICS Destination Control Table (DCT).

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) DCT(oper,oper,oper,...)
```

Prefix length

One to eight characters

Capacity of list

One to five CICS destinations per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) DCT(prefix(es))
                 ACCESS(access levels)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

DCT is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DCT ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DCT ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DCT(OWN) authority to ADD or REMOVE ownership of DCTs from ACIDs
- DCT(XAUTH) authority to permit or revoke access to DCTs

The administrator can:

- Specify any or all of the following access levels: INQUIRE, SET, ALL, FEOV, NONE, PURGE, READ, WRITE, INSTALL, nad COLLECT. If ACCESS is not specified, CA Top Secret defaults to READ access. For CICS 4.1 and above, WRITE access is required because reading a DCT record is destructive.
- Use any of the following methods to control access to DCTs: Expiration, Facility, Pathing, Time/Day, and Actions.

Examples: DCT resource class

This example protects a CICS destination by assigning ownership to the Corporate Level Division (CORPCICS):

```
TSS ADDTO(CORPCICS) DCT(PRT6)
```

This example removes ownership:

```
TSS REMOVE(CORPCICS) DCT(PRT6)
```

This example permits a user full access to the installation's DCTs:

```
TSS PERMIT(USER05) DCT(DCT1) ACCESS(ALL)
```

This example revokes access:

```
TSS REVOKE(USER05) DCT(DCT1)
```

DCTABLE Resource Class—Secure CA SYSVIEW DCTABLE

Valid on z/OS.

Use DCTABLE to secure DCTABLE for CA-CA-SYSVIEW®

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DCTABLE(`*`table`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DCTABLE(TABLE1.PRODTAB)`

Prefix length

One to forty-four characters

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DCTABLE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DCTABLE ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DCTABLE (OWN) authority to ADD or REMOVE ownership of DCTABLE resources from ACIDs
- DCTABLE(XAUTH) authority to permit or revoke access to DCTABLE resources

The administrator can use the following methods to control access to DCTABLE resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: DCTABLE resource class

This example protects the resource by adding ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) DCTABLE(TABLE1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) DCTABLE(TABLE1)
```

This example permits users in the Technical Services Department to access DCTABLE(TABLE1.PRODTAB) on Fridays only:

```
TSS PERMIT(acid) DCTABLE(TABLE1.PRODTAB) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access DCTABLE(TABLE1.PRODTAB):

```
TSS PERMIT(acid) DCTABLE(TABLE1.PRODTAB)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) DCTABLE(TABLE1.PRODTAB)
```

DEVICES Resource Class—Secure Devices

Valid on z/OS.

Use DEVICES to secure system device allocation of graphics, teleprocessing and unit record devices.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DEVICES(`*`prefix(es)`*`)`

Prefix length

two to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DEVICES(`*`device-name, device-name...`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes

DEVICES is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The DEVICES ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DEVICES ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DEVICES(OWN) authority to ADD or REMOVE device allocation authority for ACIDs
- DEVICES(XAUTH) authority to permit or revoke the ability to route messages to target userids

The DEVICES resource class supports all masking characters.

The administrator can:

- Specify the access levels NONE and READ.
- Use the following methods to control access to DEVICE resources: Expiration, Facility, Time/Day, and Actions. Only z/OS, ESA release 3.1.3 and above offer support for all resources.

Examples: DEVICES resource class

This example gives JESPGMR ownership of the JES-related unit record devices on system SYSA:

```
TSS ADDTO(JESPGMR) DEVICES(SYSA.UR)
```

This example removes ownership:

```
TSS REMOVE(JESPGMR) DEVICES(SYSA.UR)
```

This example permits ACID DRAFTING to allocate graphic devices:

```
TSS PERMIT(DRAFTING) DEVICES(ESA1.GRAPHIC) ACCESS(READ)
```

This example prevents allocation of UR type devices:

```
TSS PERMIT(ALL) DEVICES(SP01.UR) ACCESS(NONE)
```

DFTABLE Resource Class—Secure CA SYSVIEW DFTABLE

Valid on z/OS.

Use DFTABLE to secure DFTABLE for CA SYSVIEW®

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DFTABLE(`*`table`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DFTABLE(`*`table`*`)`

Prefix length

One to forty-four characters

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DFTABLE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DFTABLE ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DFTABLE (OWN) authority to ADDTO or REMOVE ownership of DFTABLE resources from ACIDs
- DFTABLE(XAUTH) authority to permit or revoke access to DFTABLE resources

The administrator can use any of the following methods to control access to DFTABLE resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: DFTABLE resource class

This example protects the resource by having it owned by the Corporate Department (owning acid):

```
TSS ADDTO(CORPORAT) DFTABLE(TABLE1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) DFTABLE(TABLE1)
```

This example permits users in the Technical Services Department to access DFTABLE(TABLE1.PRODTAB) on Fridays only:

```
TSS PERMIT(acid) DFTABLE(TABLE1.PRODTAB)
                 DAYS(FRIDAY)
```

This example permit users in the Technical Services Department to access DFTABLE(TABLE1.PRODTAB):

```
TSS PERMIT(acid) DFTABLE(TABLE1.PRODTAB)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) DFTABLE(TABLE1.PRODTAB)
```

DIAGNOSE Resource Class—Secure CP Diagnose Codes

Valid on z/VM.

Use DIAGNOSE to secure the use of CP diagnose codes.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DIAGNOSE(`*`oper`*`,...)`

Prefix length

One to four characters (0-9, A-F)

Capacity of list

One to five DIAGNOSE codes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DIAGNOSE(`*`oper`*`,...)`

Prefix length

One to four characters (0-9, A-F)

Capacity of list

One to five DIAGNOSE codes per TSS command

DIAGNOSE is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DIAGNOSE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DIAGNOSE ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DIAGNOSE(OWN) authority to ADD or REMOVE ownership of DIAGNOSE codes from ACIDs
- DIAGNOSE(XAUTH) authority to permit or revoke access to diagnose codes

The administrator can use the following methods to control access to DIAGNOSE codes: Expiration, Facility, Time/Day, and Actions.

Examples: DIAGNOSE resource class

This example protects a DIAGNOSE code 84 (Directory Update In Place), by assigning ownership to the Systems Programming Department (SYSPDEPT):

```
TSS ADDTO(SYSPDEPT) DIAGNOSE(84)
```

The administrator may now PERMIT limited access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(SYSPDEPT) DIAGNOSE(84)
```

This example permits a programmer to use DIAGNOSE code 84 (Directory Update In Place):

```
TSS PERMIT(PROGUSER) DIAGNOSE(84)
```

In this example, VMPRIV is implied.

The administrator may revoke PROGUSER's permission:

```
TSS REVOKE(PROGUSER) DIAGNOSE(84)
```

Some DIAGNOSE codes have both privileged and nonprivileged subcodes; therefore, This example permits an ACID to execute nonprivileged forms of DIAGNOSE 80 (MSSFCALL):

```
TSS PERMIT(USER01) DIAGNOSE(80)
```

This example permits USER01 to issue privileged forms of DIAGNOSE 80 (for example, to write to the IOCDS):

```
TSS PERMIT(USER01) DIAGNOSE(80)
                   ACTION(VMPRIV)
```

DIRECTRY Resource Class—Secure DIRECTRY for TSSVM

Valid on z/VM.

Use DIRECTRY to secure DIRECTRY for TSSVM.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) DIRECTRY(DIRECTRY1)
```

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) DIRECTRY(DIRECTRY1)
```

Prefix length

One to eight characters

DIRECTRY is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DIRECTRY ACID types User, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DIRECTRY ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DIRECTRY (OWN) authority to ADDTO or REMOVE ownership of DIRECTRY resources from ACIDs
- DIRECTRY(XAUTH) authority to permit or revoke access to DIRECTRY resources

The administrator can use the following methods to control access to DIRECTRY resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: DIRECTRY resource class

This example protects the resource by assigning ownership to the Corporate Department (owning acid):

```
TSS ADDTO(CORPORAT) DIRECTRY(DIRECTRY1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) DIRECTRY(DIRECTRY1)
```

This example permits users to access DIRECTRY(DIRECTRY1) on Fridays only:

```
TSS PERMIT(TECHUSER) DIRECTRY(DIRECTRY1)
                     DAYS(FRIDAY)
```

This example permit users in the Technical Services Department to access DIRECTRYR1.ABCDEFGH:

```
TSS PERMIT(TECHPROF) DIRECTRY(DIRECTRY1)
```

This example revokes access:

```
TSS REVOKE(TECHUSER) DIRECTRY(DIRECTRY1)
```

DLFCLASS Resource Class—Determine DLF Record Access

Valid on z/OS.

Use DLFCLASS to determine which users can access DLF record.

Under Release 3.1.3 of z/OS, ESA, the Data Lookaside Facility (DLF) is used to control the loading of data sets into ESA hyperspace by selected jobs. These data sets are identified to CA Top Secret by adding them to the DLF record. Once they've been identified, the DLFCLASS resource class keyword is used to determine which users can access these data sets. For information about the DLF record, see the *Implementation: BATCH and STC Guide*.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) DLFCLASS(oper,...)
```

Prefix length

two to twenty-six characters

Capacity of list

One to five data set names or prefixes per TSS command

Generic prefixing allows the administrator to group a set of similar data sets together, and define them by one generic prefix.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) DLFCLASS(prefix(es))
                 ACCESS(access levels)
```

Prefix length

Two to forty-four characters

Capacity of list

One to five data set names, prefixes, or masks per TSS command

DLFCLASS is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DLFCLASS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DLFCLASS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DLFCLASS(OWN) authority to ADD or REMOVE ownership of DLF data sets from ACIDs

Note: A fully qualified DLF data set name is PERMITted to an ACID by enclosing in single quotation marks. This will indicate that it is defined to CA Top Secret, not as a prefix, but by its fully qualified name.

Data set masking is another method of reducing the number of DLF data set definitions to implement widespread DLF data set protection.

The administrator can:

- Use any of the following methods to control access to DLF data sets: Expiration, Facility, Program Pathing, Time/Day, and Actions.
- Specify the access levels: ALL, CREATE, CONTROL, FETCH, NONE, READ, SCRATCH, UPDATE, and WRITE. If ACCESS is not specified, CA Top Secret defaults to READ access.

Example: DLFCLASS resource class

This example gives the Inventory Department (INVDEPT) ownership of the DLF data set known as CICS.MSTR.FILE:

```
TSS ADDTO(INVDEPT) DLFCLASS(CICS.MSTR.FILE)
```

This example removes ownership:

```
TSS REMOVE(INVDEPT) DLFCLASS(CICS.MSTR.FILE)
```

This example permits USER01 to access the CICS.MSTR.FILE DLF data set:

```
TSS PERMIT(USER01) DLFCLASS(CICS.MSTR.FILE)
```

Before issuing this command, verify that CICS.MSTR.FILE has been added to the DLF record.

This example revokes USER01's access to this DLF data set:

```
TSS REVOKE(USER01) DLFCLASS(CICS.MSTR.FILE)
```

DRTABLE Resource Class—Secure CA SYSVIEW DRTABLE

Valid on z/OS.

Use DRTABLE to secure DRTABLE for CA SYSVIEW®.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) DRTABLE(TABLE1)
```

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) DRTABLE(TABLE1.PRODTAB)
```

Prefix length

One to 44 characters

DRTABLE is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DRTABLE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DRTABLE ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DRTABLE (OWN) authority to ADD or REMOVE ownership of DRTABLE resources from ACIDs
- DRTABLE(XAUTH) authority to permit or revoke access to DRTABLE resources

The administrator can use any of the following methods to control access to DRTABLE resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: DRTABLE resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department (owning acid):

```
TSS ADDTO(CORPORAT) DRTABLE(TABLE1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) DRTABLE(TABLE1)
```

This example permits users in the Technical Services Department to access DRTABLE(TABLE1.PRODTAB) on Fridays only:

```
TSS PERMIT(acid) DRTABLE(TABLE1.PRODTAB)
                 DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access DRTABLE(TABLE1.PRODTAB):

```
TSS PERMIT(acid) DRTABLE(TABLE1.PRODTAB)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) DRTABLE(TABLE1.PRODTAB)
```

DSNAME Resource Class—Secure Data Sets

Valid on z/OS, z/VSE, and z/VM.

Use DSNAME to secures data sets. You can identify each data set separately or in groups of like-named data sets (using generic prefixing and masking techniques).

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) DSNAME(oper,...)
```

Prefix length

Two to twenty-six characters

Capacity of list

One to five data set names or prefixes per TSS command

Generic prefixing allows the administrator to group a set of similar data sets together, and define them by one generic prefix.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) DSNAME(prefix(es))
                 ACCESS(access levels)
```

Prefix length

Two to forty-four characters

Capacity of list

One to five data sets names, prefixes, or masks per TSS command

Note: A fully qualified data set name is PERMITted to an ACID by enclosing in single quotation marks. This will indicate that it is defined to CA Top Secret, not as a prefix, but by its fully qualified name.

Data set masking is another method of reducing the number of data set definitions to implement widespread data set protection.

DSNAME is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DSNAME ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DSNAME ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DSNAME(OWN) authority to ADD or REMOVE ownership of data sets from ACIDs

The administrator can:

- Use the following methods to control access to data sets: Expiration, Facility, Program Pathing, Time/Day, and Actions
- Specify the access levels: ALL, CREATE, CONTROL, FETCH, NONE, READ, SCRATCH, UPDATE, and WRITE. If ACCESS is not specified, CA Top Secret defaults to READ

Example: DSNAME resource class

This example gives the Inventory Department (INVDEPT) ownership of a data set known as UNSOLD.INV.MASTER.FILE:

```
TSS ADDTO(INVDEPT) DSNAME('UNSOLD.INV.MASTER.FILE')
```

This example removes ownership:

```
TSS REMOVE(INVDEPT) DSNAME('UNSOLD.INV.MASTER.FILE')
```

This example has USER01 access any data set prefixed by SFT:

```
TSS PERMIT(USER01) DSNAME(SFT.)
```

This eliminates the need to permit access to 'SFT.IMS.PROD' and 'SFT.IMS.SCEDS'.

This example revokes USER01's access to all data sets beginning with the prefix SFT:

```
TSS REVOKE(USER01) DSNAME(SFT.)
```

DSPACE Resource Class—Restrict Access to Databases

Valid on z/VM (VM/ESA Release 2.0 and above only).

Use the DSAPCE to restrict access to a virtual machine's data spaces.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) DSPACE(oper.,oper.,...)
```

Prefix length

Two to 26 characters

Capacity of list

One to five data spaces per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) DSPACE(prefix(es)) ACCESS(levels)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command.

This resource class is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- DSPACE(OWN) authority

The administrator can:

Specify the access levels ALL, READ, UPDATE, or NONE. The default is READ.

Use the following methods to control access to DSPACE resources Expiration, Facility, Program Pathing, Time/Day, and Actions

Examples: DSPACE resource class

This example assigns ownership of the SFSESA20 machine to the Corporate Department:

```
TSS ADD(CORPDEPT) DSPACE(SFSESA2)
```

This example removes ownership:

```
TSS REMOVE(CORPDEPT) DSPACE(SFSESA2)
```

This example permits users in the Technical Services Department to access SFSESA20:

```
TSS PERMIT(TECHPROF) DSPACE(SFSESA2)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) DSPAC(SFSESA2)
```

DSTABLE Resource Class—Secure CASYSVIEW DSTABLE

Valid on z/OS.

Use DSTABLE to secure DSTABLE for CA-SYSVIEW.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DSTABLE(`*`table`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DSTABLE(`*`table`*`)`

Prefix length

One to forty-four characters

DSTABLE is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DSTABLE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DSTABLE ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DSTABLE (OWN) authority to ADD or REMOVE ownership of DSTABLE resources from ACIDs
- DSTABLE(XAUTH) authority to permit or revoke access to DSTABLE resources owned

The administrator can use any of the following methods to control access to DSTABLE resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: DSTABLE resource class

This example protects a resource using CA Top Secret by assigning ownership to the Corporate Department (owning acid):

```
TSS ADDTO(CORPORAT) DSTABLE(TABLE1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) DSTABLE(TABLE1)
```

This example permits users in the Technical Services Department to access DSTABLE(TABLE1.PRODTAB) on Fridays only:

```
TSS PERMIT(acid) DSTABLE(TABLE1.PRODTAB)
                 DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access DSTABLE(TABLE1.PRODTAB):

```
TSS PERMIT(acid) DSTABLE(TABLE1.PRODTAB)
```

This example revokes acces:

```
TSS REVOKE(TECHPROF) DSTABLE(TABLE1.PRODTAB)
```

DTADMIN Resource Class—Secure CASYSVIEW DTADMIN

Valid on z/OS.

Use DTADMIN to secures DTADMIN for CA-SYSVIEW.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*acid*`) DTADMIN(`*admin*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`) DTADMIN(`*admin*`)`

Prefix length

One to forty-four characters

DTADMIN is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DTADMIN ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DTADMIN ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DTADMIN (OWN) authority to ADD or REMOVE ownership of DTADMIN resources from ACIDs
- DTADMIN(XAUTH) authority to permit or revoke access to DTADMIN resources

The administrator can use any of the following methods to control access to DTADMIN resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: DTADMIN resource class

This example protects resources by with CA Top Secret by assigning ownership to the Corporate Department (owning acid):

```
TSS ADDTO(CORPORAT) DTADMIN(ADMIN1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) DTADMIN(ADMIN1)
```

This example permits users in the Technical Services Department to access DTADMIN(ADMIN1.PRODADM) on Fridays only:

```
TSS PERMIT(TECHPROF) DTADMIN(ADMIN1.PRODADM)
                DAYS(FRIDAY)
```

This example permit users in the Technical Services Department to access DTADMIN(ADMIN1.PRODADM):

```
TSS PERMIT(TECHPROF) DTADMIN(ADMIN1.PRODADM)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) DTADMIN(ADMIN1.PRODADM)
```

DTSYSTEM Resource Class—Secure CASYSVIEW DTSYSTEM

Valid on z/OS.

Use DTSYSTEM to secure DTSYSTEM for CA SYSVIEW.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*acid*`) DTSYSTEM(`*system*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`) DTSYSTEM(`*system*`)`

Prefix length

One to 44 characters

DTSYSTEM is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS.
- The DTSYSTEM ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- The DTSYSTEM ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE when used with TSS ADD/REMOVE
- DTSYSTEM (OWN) authority to ADD or REMOVE ownership of DTSYSTEM resources from ACIDs
- DTSYSTEM(XAUTH) authority to permit or revoke access to DTSYSTEM resources

The administrator can use any of the following methods to control access to DTSYSTEM resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: DTSYSTEM resource class

This example protects the resource using CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) DTSYSTEM(SYSTEM1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) DTSYSTEM(SYSTEM1)
```

This example permit users in the Technical Services Department to access DTSYSTEM(SYSTEM1.PRODSYS) on Fridays only:

```
TSS PERMIT(TECHPROF) DTSYSTEM(SYSTEM1.PRODSYS)
                DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access DTSYSTEM(SYSTEM1.PRODSYS)

```
TSS PERMIT(TECHPROF) DTSYSTEM(SYSTEM1.PRODSYS)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) DTSYSTEM(SYSTEM1.PRODSYS)
```

DTTABLE Resource Class—Secure CASYSVIEW DTTABLE

Valid on z/OS.

Use DTTABLE to secure DTTABLE for CA-SYSVIEW.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*acid*`) DTTABLE(`*table*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`) DTTABLE(`*table*`)`

Prefix length

One to forty-four characters

DTTABLE is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DTTABLE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- The DTTABLE ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE when used with TSS ADD/REMOVE
- DTTABLE (OWN) authority to ADD or REMOVE ownership of DTTABLE resources from ACIDs
- DTTABLE(XAUTH) authority to permit or revoke access to DTTABLE resources

The administrator can use any of the following methods to control access to DTTABLE resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: DTTABLE resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) DTTABLE(TABLE1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) DTTABLE(TABLE1)
```

This example permit users in the Technical Services Department to access DTTABLE(TABLE1.PRODTAB) on Fridays only:

```
TSS PERMIT(TECPROF) DTTABLE(TABLE1.PRODTAB) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access DTTABLE(TABLE1.PRODTAB)

```
TSS PERMIT(TECHPROF) DTTABLE(TABLE1.PRODTAB)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) DTTABLE(TABLE1.PRODTAB)
```

DTUTIL Resource Class—Secure CASYSVIEW DTUTIL

Valid on z/OS.

Use DTUTIL to secure DTUTIL for CA SYSVIEW.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DTUTIL(`*`utility`*`)`

Prefix length

One to eight characters.

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DTUTIL(`*`utility`*`)`

Prefix length

One to 44 characters

DTUTIL is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DTUTIL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DTUTIL ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DTUTIL (OWN) authority to ADD or REMOVE ownership of DTUTIL resources from ACIDs
- DTUTIL(XAUTH) authority to permit or revoke access to DTUTIL resources

The administrator can use any of the following methods to control access to DTUTIL resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: DTUTIL resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) DTUTIL(UTILITY1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) DTUTIL(UTILITY1)
```

This example permits users in the Technical Services Department to access DTUTIL(UTILITY1.PRODUTIL) on Fridays only:

```
TSS PERMIT(TECHPROF) DTUTIL(UTILITY1.PRODUTIL) DAYS(FRIDAY)
```

This example permit users in the Technical Services Department to access DTUTIL(UTILITY1.PRODUTIL):

```
TSS PERMIT(TECHPROF) DTUTIL(UTILITY1.PRODUTIL)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) DTUTIL(UTILITY1.PRODUTIL)
```

DXTABLE Resource Class—Secure CASYSVIEW DXTABLE

Valid on z/OS.

Use DXTABLE to secures DXTABLE for CA SYSVIEW.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) DXTABLE(`*`table`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) DXTABLE(`*`table`*`)`

Prefix length

One to 44 characters

DXTABLE is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The DXTABLE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The DXTABLE ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- DXTABLE (OWN) authority to ADD or REMOVE ownership of DXTABLE resources from ACIDs
- DXTABLE(XAUTH) authority to permit or revoke access to DXTABLE resources

The administrator can use any of the following methods to control access to DXTABLE resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Example: DXTABLE resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) DXTABLE(TABLE1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) DXTABLE(TABLE1)
```

This example permits users in the Technical Services Department to access DXTABLE(TABLE1.PRODTAB) on Fridays only:

```
TSS PERMIT(TECHPROF) DXTABLE(TABLE1.PRODTAB)
                DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access DXTABLE(TABLE1.PRODTAB):

```
TSS PERMIT(TECHPROF) DXTABLE(TABLE1.PRODTAB)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) DXTABLE(TABLE1.PRODTAB)
```

EJBROLE Resource Class—Secure z/OS Enterprise Beans

Valid on z/OS.

Use the EJBROLE or GEJBROLE classes in CA Top Secret to control access to enterprise beans for a client.

The administrator can use any of the following methods to control access to EJBROLE and GEJBROLE resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

The administrator can use Expiration, Facility, Program Pathing, Time/Day, and Actions methods to control access to APPCLU resources.

When used with the ADDTO/REMOVE commands, this resource class has the following format:

```
TSS REMOVE(acidname) EJBROLE(rolename)
TSS REMOVE(acidname) GEJBROLE(rolename)
```

Prefix length

2 to 26 characters.

Capacity of list

One to eight prefixes per TSS command.

When used with the PERMIT/REVOKE commands, this resource class has the following format:

```
TSS REVOKE(acidname) EJBROLE(rolename)
TSS REMOVE(acidname) GEJBROLE(rolename)
```

Prefix length

2 to 245 characters

Capacity of list

One to eight prefixes per command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS.
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- EJBROLE (OWN) authority to ADD or REMOVE ownership of EJBROLE resources

- EJBROLE(XAUTH) authority to PERMIT or REVOKE access to EJBROLE resources
- GEJBROLE (OWN) authority to ADD or REMOVE ownership of GEJBROLE resources
- GEJBROLE(XAUTH) authority to PERMIT or REVOKE access to GEJBROLE resources

Examples:

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid). The administration can then PERMIT access to users or profiles that require access.

```
TSS ADDTO(CORPORAT) GEJBROLE(ROLE)

TSS ADDTO(CORPORAT) EJBROLE(ROLE)
```

This example permits users in the Technical Services Department to access GEJBROLE(ROLE2):

```
TSS PERMIT(TECHPROF) GEJBROLE(ROLE2)
```

This example permits users in the Technical Services Department to access EJBROLE(ROLE1) on Fridays only:

```
TSS PERMIT(TECHPROF) EJBROLE(ROLE1) DAYS(FRIDAY)
```

FCT Resource Class—Secure CICS File Control Table Entries

Valid on z/OS and z/VSE.

Use FCT to secures File Control Table entries (DDnames) in CICS.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) FCT(oper,oper,oper,...)
```

Prefix length

One to eight characters

Capacity of list

One to five FCT prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) FCT(prefix(es))
                 ACCESS(access levels)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The FCT ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The FCT ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- FCT(OWN) authority to ADD or REMOVE ownership of FCTs from ACIDs
- FCT(XAUTH) authority to permit or revoke access to FCTs

The administrator can:

- Specify the access levels: SET, INQUIRE, ALL, BROWSE, DELETE, NONE, READ, UPDATE, WRITE, INSTALL, COLLECT. If ACCESS is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to FCTs: Expiration, Facility, PPT Pathing, Time/Day, and Actions.

Examples: FCT resource class

This example assigns ownership of an FCT prefix to a department ACID:

```
TSS ADDTO(PAYDEPT) FCT(PAY)
```

This example removes ownership:

```
TSS REMOVE(PAYDEPT) FCT(PAY)
```

This example permits a group of users to read and browse two CICS files:

```
TSS PERMIT(PERPROF) FCT(SKILLS,PENSION)
                    ACCESS(READ,BROWSE)
```

This example revokes access:

```
TSS REVOKE(PERPROF) FCT(SKILLS,PENSION)
```

FIELD Resource Class—Secure Database Fields

Valid on z/OS and z/VSE.

Use FIELD to secure installation-defined database fields.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) FIELD(oper,oper,oper,...)
```

Prefix length

One to eight characters

Capacity of list

One to five FIELD prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) FIELD(prefix(es))
                 ACCESS(access levels)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The FIELD ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The FIELD ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- FIELD(OWN) authority to ADD or REMOVE ownership of FIELDs from ACIDs
- FIELD(XAUTH) authority to permits or revoke access to FIELDs that are owned

The administrator can:

- Specify the access levels: ALL, NONE, READ, UPDATE, and WRITE. If ACCESS is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to FIELDs: Expiration, Facility, Pathing, Time/Day, Actions.

Protection

Fields are not automatically protected by CA Top Secret, even when defined via the ADDTO command function. An application program must perform the security check by calling CA Top Secret via FRACHECK or the application high-level language interface.

Examples: FIELD resource class

This example protects a group of fields for a CICS application:

```
TSS ADDTO(DEPTC) FIELD(A100,A200,A300)
```

This example removes ownership:

```
TSS REMOVE(DEPTC) FIELD(A100,A200,A300)
```

This example permits CICS clerks to update fields daily from 8:00 a.m. until 5:00 p.m., but only through a specific CICS application program (on behalf of a transaction):

```
TSS PERMIT(PRFCLRK) FIELD(A1006,A3002)
                    TIMES(08,17)
                    DAYS(WEEKDAYS)
                    PRIVPGM(ISPTASK1)
                    ACCESS(U)
```

This example revokes access:

```
TSS REVOKE(PRFCLRK) FIELD(A1006,A3002)
```

FSACCESS Resource Class—Secure zFS File System Container

Valid on z/OS.

Use FSACCESS to secure access to a zFS file system container (that is, a data set).

When used with TSS ADDTO or REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) FSACCESS(`*`dataset`*`)`

Prefix length

2-26 characters

Capacity of list

1-8 prefixes per TSS command

When used with TSS PERMIT or REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) FSACCESS(`*`dataset`*`)`

Prefix length

1-44 characters

Capacity of list

1-8 prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The FSACCESS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD or REMOVE
- The FSACCESS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT or REVOKE
- FSACCESS(OWN) authority to ADD or REMOVE ownership of FSACCESS resources from ACIDs
- FSACCESS(XAUTH) authority to permit or revoke access to FSACCESS resources

The administrator can use any of the following methods to control access to FSACCESS resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: FSACCESS resource class

This example protects the resource by assigning ownership to the Corporate Department. After ownership is assigned, the administrator can PERMIT access to users or profiles.

```
TSS ADDTO(CORPORAT) FSACCESS(ZFS)
```

This example permits users in the Technical Services Department to access dataset ZFS.DATA.CON1:

```
TSS PERMIT(TECHPROF) FSACCESS(ZFS.DATA.CON1)
```

This example revokes access to data set ZFS.DATA.CON1:

```
TSS REVOKE(TECHPROF) FSACCESS(ZFS.DATA.CON1)
```

GEJBROLE Resource Class—Secure z/OS Enterprise Beans

Valid on z/OS.

Use the EJBROLE or GEJBROLE classes in CA Top Secret to control access to enterprise beans for a client.

The administrator can use any of the following methods to control access to EJBROLE and GEJBROLE resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

The administrator can use Expiration, Facility, Program Pathing, Time/Day, and Actions methods to control access to APPCLU resources.

When used with the ADDTO/REMOVE commands, this resource class has the following format:

```
TSS REMOVE(acidname) EJBROLE(rolename)
TSS REMOVE(acidname) GEJBROLE(rolename)
```

Prefix length

2 to 26 characters.

Capacity of list

One to eight prefixes per TSS command.

When used with the PERMIT/REVOKE commands, this resource class has the following format:

```
TSS REVOKE(acidname) EJBROLE(rolename)
TSS REMOVE(acidname) GEJBROLE(rolename)
```

Prefix length

2 to 245 characters

Capacity of list

One to eight prefixes per command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS.
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- EJBROLE (OWN) authority to ADD or REMOVE ownership of EJBROLE resources

- EJBROLE(XAUTH) authority to PERMIT or REVOKE access to EJBROLE resources
- GEJBROLE (OWN) authority to ADD or REMOVE ownership of GEJBROLE resources
- GEJBROLE(XAUTH) authority to PERMIT or REVOKE access to GEJBROLE resources

Examples:

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid). The administration can then PERMIT access to users or profiles that require access.

```
TSS ADDTO(CORPORAT) GEJBROLE(ROLE)

TSS ADDTO(CORPORAT) EJBROLE(ROLE)
```

This example permits users in the Technical Services Department to access GEJBROLE(ROLE2):

```
TSS PERMIT(TECHPROF) GEJBROLE(ROLE2)
```

This example permits users in the Technical Services Department to access EJBROLE(ROLE1) on Fridays only:

TSS PERMIT(TECHPROF) EJBROLE(ROLE1) DAYS(FRIDAY)

GSOMDOBJ Resource Class—Secure z/OS GSOMDOBJ Objects

Valid on z/OS.

Use GSOMDOBJ to secure GSOMDOBJ for z/OS objects

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) GSOMDOBJ(`*`object`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) GSOMDOBJ(`*`object`*`)`

Prefix length

One to sixteen characters

Capacity of list

One to eight prefixes per command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The GSOMDOBJ ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The GSOMDOBJ ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- GSOMDOBJ (OWN) authority to ADD or REMOVE ownership of GSOMDOBJ resources from ACIDs
- GSOMDOBJ(XAUTH) authority to permit or revoke access to GSOMDOBJ resources

The administrator can use any of the following methods to control access to GSOMDOBJ resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: GSOMDOBJ resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) GSOMDOBJ(OBJECT1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) GSOMDOBJ(OBJECT1)
```

This example permits users in the Technical Services Department to access OBJECT1.XXXXXXXX on Fridays only:

```
TSS PERMIT(TECHPROF) GSOMDOBJ(OBJECT1.XXXXXXXX)
                     DAYS(FRIDAY)
```

This example permits users in Technical Services Department to access OBJECT1.XXXXXXXX:

```
TSS PERMIT(TECHPROF) GSOMDOBJ(OBJECT1.XXXXXXXX)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) GSOMDOBJ(OBJECT1.XXXXXXXX)
```

HFSSEC Resource Class—Secure HFS Files

Valid on z/OS.

Use HFSSEC to secure HFS files under CA Top Secret. For information on HFSSEC, see the *Cookbook*.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) HFSSEC(/nnnnnn)
```

Prefix length

1 to 8 characters

Capacity of list

1 to 8 prefixes per command

Note: When you add HFSSEC, add a slash (/) unless you are using ROOT. Administratively, you can ADD and PERMIT HFSSEC without using a slash, but functionally it will not work.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) HFSSEC(/nnnnnn)
                 ACCESS(READ)
```

Prefix length

1 to 255 characters

Note: When you permit HFSSEC, add a slash (/) unless you are using ROOT. Administratively, you can ADD and PERMIT HFSSEC without using a slash, but functionally it will not work.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The HFSSEC ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The HFSSEC ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- HFSSEC (OWN) authority to ADD or REMOVE ownership of HFSSEC resources from ACIDs
- HFSSEC(XAUTH) authority to permit or revoke access to HFSSEC resources

The administrator can use any of the following methods to control access to HFSSEC resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: HFSSEC resource class

This example protects the resource by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) HFSSEC(ROOT)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) HFSSEC(ROOT)
```

This example permits users in the Technical Services Department to access BIN on Fridays only:

```
TSS PERMIT(TECHPROF) HFSSEC(/BIN)
                     ACCESS(READ,EXEC)
                     DAYS(FRIDAY)
```

This example permit users in the Technical Services Department to access HFSSEC(/BIN):

```
TSS PERMIT(TECHPROF) HFSSEC(/BIN)
                     ACCESS(READ,EXEC)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) HFSSEC(/BIN)
                     ACCESS(READ,EXEC)
```

IBMFAC Resource Class—Determine IBM Facilities Ownership

Valid on z/OS and z/VSE.

Use IBMFAC determine who has ownership or control over catalog, IDCAMS, SMS, DFDSS and other IBM Facilities.

This resource class is the equivalent to the FACILITY resource class in RACF.

Important! CA Top Secret also supports the XFACILIT and GXFACILIT resource classes, which support prefix lengths up to 246 characters. These resource classes follow the same usage rules as IBMFAC. For detailed information on these resource classes, see the IBM documentation.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) IBMFAC(`*`facility`*`,...)`

Prefix length

One to eight characters

Capacity of list

Five facilities per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) IBMFAC(`*`facility`*`,...)`

Prefix length

One to forty-four characters

Capacity of list

Five facilities per TSS command

This keyword uses:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The IBMFAC ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The IBMFAC ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- IBMFAC(OWN) authority to protect user access to IBM facilities.
- IBMFAC(XAUTH) authority to permit or revoke access to IBM facilities

The administrator can:

- Use any of the following methods to control access to IBMFAC: Expiration, Facility, Time/Day, and Actions.
- Specify the access levels: ALL, NONE, READ, UPDATE, and WRITE. If ACCESS is not specified, CA Top Secret defaults to NONE access.

Examples: IBMFAC resource class

This example adds an IBMFAC of IDC.DIAG:

```
TSS ADDTO(DEPT02) IBMFAC(IDC.DIAG)
```

This example removes ownership:

```
TSS REMOVE(DEPT02) IBMFAC(IDC.DIAG)
```

This example allows an ACID to use IBMFAC:

```
TSS PERMIT(DEPT02) IBMFAC(IDC.DIAG)
```

This example revokes access:

```
TSS REVOKE(DEPT02) IBMFAC(IDC.DIAG)
```

This example grants USER01 control access to BPX.Example IBM facility:

```
TSS PERMIT(USER01) IBMFAC(BPX.EXAMPLE) ACCESS(CONTROL)
```

IBMGROUP Resource Class—Assign Group Names

Valid on z/OS.

Use IBMGROUP at signon to assign group names to users. For information about signon processing, see the *User Guide*. The group names specified are later queried by such products as DB2 and DF/HSM.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) IBMGROUP(`*`group`*`,..|`*`authid`*`,...)`

Prefix length

One to eight characters

Capacity of list

five resources per TSS command

Note: Under CA Top Secret for DB2, the IBMGROUP resource class is used in reference to secondary authorization IDs.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) IBMGROUP(`*`group`*`,...|`*`authid`*`,...)`

Prefix length

One to eight characters

Capacity of list

five resources per TSS command

Note: Under CA Top Secret for DB2, the IBMGROUP resource class is used in reference to secondary authorization IDs.

This keyword uses:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The IBMGROUP ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- IBMGROUP(OWN) authority to ADD or REMOVE ownership of IBM type group names
- IBMGROUP(XAUTH) authority to permit or revoke access to resources

The administrator can use any of the following methods to control access to IBMGROUP: Expiration, Facility, Time/Day, and Actions. The access controls CALENDAR, TIMEREC, SYSID may not be used with this resource.

Examples: IBMGROUP resource class

This example protects the use of DB2 PAYROLL secondary authorization ID, by assigning ownership to the Department ACID, and subsequently permitting restricted access to users or profiles:

```
TSS ADDTO(DEPT02) IBMGROUP(PAYROLL)
```

This example removes ownership:

```
TSS REMOVE(DEPT02) IBMGROUP(PAYROLL)
```

This example permits use of the PAYROLL secondary authorization ID from Batch only:

```
TSS PERMIT(SYSAU2) IBMGROUP(PAYROLL)
                   FACILITY(BATCH)
```

This example revokes access:

```
TSS REVOKE(SYSAU2) IBMGROUP(PAYROLL)
```

IUCV Resource Class—Secure IUCV Target Users

Valid on z/VM.

Use IUCV to secure Inter User Communication Vehicle (IUCV) target users.

Note: IUCV communication targets controls the ability of users to issue an IUCV CONNECT to the virtual machine designated by the resource name.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) IUCV(`*`userid`*`,`*`userid`*`,`*`userid`*`,...)`

Prefix length

One to eight characters

Capacity of list

One to five userids per TSS command

Generic prefixing allows the administrator to group a set of similar resource names together, and define them by one generic prefix.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) IUCV(`*`oper`*`,`*`oper`*`,`*`oper`*`,...)`

Prefix length

One to eight characters

Capacity of list

Five resources per TSS command

This keyword uses:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The IUCV ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The IUCV ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- IUCV(OWN) authority to ADD or REMOVE ownership from ACIDs
- IUCV(XAUTH) authority to permit or revoke access to resources

The administrator can use any of the following methods to control access to IUCV: Expiration, Time/Day, and Actions.

Examples: IUCV resource class

This example protects users in the application department issuing an IUCV connect to virtual machine APP17:

```
TSS ADDTO(APPDEPT) IUCV(APP17)
```

This example removes ownership:

```
TSS REMOVE(APPDEPT) IUCV(APP17)
```

This example permits user VMUSER1 to connect with DBSERVE via IUCV:

```
TSS PERMIT(VMUSER1) IUCV(DBSERVE)
```

This example revokes access:

```
TSS REVOKE(VMUSER1) IUCV(DBSERVE)
```

JCT Resource Class—Secure JCT Entries

Valid on z/OS and z/VSE

Use JCT to secure entries in the CICS Journal Control Table (JCT).

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) JCT(oper,oper,oper,...)
```

Prefix length

One to seventeen characters

Capacity of list

One to five JCT prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) JCT(prefix(es))
                 ACCESS(access levels)
```

Prefix length

One to eight characters

Capacity of list

One to five JCT prefixes per TSS command

This keyword uses:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The JCT ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The JCT ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- JCT(OWN) authority to ADD or REMOVE ownership of JCTs from ACIDs
- JCT(XAUTH) authority to permit or revoke access to resources

The administrator can:

- Specify the access levels: INQUIRE, SET, ALL, NONE, WRITE. If ACCESS is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to JCTs: Expiration, Facility, PPT Pathing, Time/Day, and Actions.

Example: JCT resource class

This example assigns ownership of a JCT, by adding the JCT to Department 1:

```
TSS ADDTO(DEPT1) JCT(J03)
```

This example removes ownership:

```
TSS REMOVE(DEPT1) JCT(J03)
```

This example permits a user all access to a predefined journal through the CICSTEST facility:

```
TSS PERMIT(GKM75) JCT(JC03)
                  ACCESS(ALL)
                  FACILITY(CICSTEST)
```

This example revokes access:

```
TSS PERMIT(GKM75) JCT(JC03)
```

JESINPUT Resource Class—Secure JES Job Entry Sources

Valid on z/OS.

Use JESINPUT to secure JES job entry sources checked during job initiation.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) JESINPUT(`*`srcnode`*`)`

Prefix length

Two to twenty-six characters

Capacity of list

One to eight prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) JESINPUT(`*`srcnode`*`)`

Prefix length

Two to forty-four characters

Capacity of list

One to eight prefixes per command

This keyword uses:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The JESINPUT ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The JESINPUT ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- JESINPUT(OWN) authority to ADD or REMOVE access for JES job sources for ACIDs
- JESINPUT(XAUTH) authority to permit or revoke access to JESINPUT sources

The administrator can:

- Specify the access levels: ALL, NONE, READ, UPDATE, and CONTROL. If ACCESS is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to JESINPUT: Expiration, Facility, Time/Day, and Actions.

Note: Only z/OS, ESA release 3.1.3 and above offer support for all resources.

Masking

Masking is not supported for JESINPUT.

Examples: JESINPUT resource class

This example adds ownership for USER01:

```
TSS ADDTO(USER01) JESINPUT(srcnode)
```

This example removes ownership:

```
TSS REMOVE(USER01) JESINPUT(srcnode)
```

JESJOBS Resource Class—Secure Jobs

Valid on z/OS.

Use JESJOBS to secure the submission and cancellation of jobs.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) JESJOBS(`*`oper`*`,`*`oper`*`...)`

Prefix length

Two to twenty-six characters

Capacity of list

One to eight prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) JESJOBS(`*`prefix(es)`*`)`

Prefix length

Two to forty-four characters

Capacity of list

One to eight prefixes per TSS command

This keyword uses:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The JESJOBS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The JESJOBS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- JESJOBS(OWN) authority to ADD or REMOVE authority to submits jobs for ACIDs
- JESJOBS(XAUTH) authority to permit or revoke access to a source validation

The administrator can:

- Specify the access levels: NONE, READ, WRITE, UPDATE, CONTROL, and ALL. If ACCESS is not specified, CA Top Secret defaults to READ access
- Use any of the following methods to control access to JESJOBS: Expiration, Facility, Time/Day, and Actions.

Note: Only z/OS, ESA release 3.1.3 and above offer support for all resources.

Masking

The JESJOBS resource class supports all masking characters.

Examples: JESJOBS resource class

This example gives an ACID, USER01, control of the jobname submitted during job submission:

```
TSS ADDTO(USER01) JESJOBS(SUBMIT.MYNODE.JOB01)
```

This example removes ownership:

```
TSS REMOVE(USER01) JESJOBS(SUBMIT.MYNODE.JOB01)
```

This example permits USER01 control of another ACIDs jobname during job submission:

```
TSS PERMIT(USER01) JESJOBS(SUBMIT.MYNODE.*.USER02)
```

This example revokes access the administrator enter:

```
TSS REVOKE(USER01) JESJOBS(SUBMIT.MYNODE.JOB2.USER02)
```

JESSPOOL Resource Class—Secure JES Spool Data Sets

Valid on z/OS.

Use JESSPOOL to secures access to the JES spool data sets, both SYSIN and SYSOUT. JESSPOOL only applies to JES2 SP3.1.3 or higher, and JES3 SP3.1.3 or higher.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) JESSPOOL(`*`oper`*`,`*`oper`*`...)`

Prefix length

Two to twenty-six characters

Capacity of list

One to eight prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) JESSPOOL(`*`prefix(es)`*`)`

Prefix length

One to fifty-three characters

Capacity of list

One to eight prefixes per TSS command

This keyword uses:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The JESSPOOL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The JESSPOOL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- JESSPOOL(OWN) authority to ADD or REMOVE access of JES spool data sets from ACIDs
- JESSPOOL(XAUTH) authority to permit or revoke access to JES spool data sets

The administrator can:

- Specify the access levels: ALL, NONE, READ, CONTROL, and UPDATE. If ACCESS is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to JESSPOOL: Expiration, Facility, Time/Day, and Actions.

Note: Only z/OS, ESA release 3.1.3 and above offer support for all resources.

Masking

The JESSPOOL resource class supports all masking characters.

Examples: JESSPOOL resource class

This example gives USER01 ownership of all JES spool data sets originating at node a, which is being routed through that node to another node (node b). The resource name that JES is validating is localnodeid.userid.jobname.jobid.dsnumber.dsname:

```
TSS ADDTO(USER01) JESSPOOL(USG203ME)
```

This example removes ownership:

```
TSS REMOVE(USER01) JESSPOOL(USG203ME)
```

This example permits users READ access of all of their own JES spool data sets from the node USG203ME:

```
TSS PERMIT(ALL) JESSPOOL(USG203ME.%)
```

This example revokes access:

```
TSS REVOKE(ALL) JESSPOOL(USG203ME.%)
```

A user needs authorization to view his output regardless of the originating node:

```
TSS PERMIT(ALL) JESSPOOL(*.%)
```

where the first qualifier (node) is any node but the second qualifier must match the userid.

JOBNAME Resource Class—Secure Jobname

Valid on z/OS.

Use JOBNAME to secure JOBNAME if site is doing a call for JOBNAME.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) JOBNAME(`*`jobname`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) JOBNAME(`*`jobname`*`)`

Prefix length

One to eight characters

This keyword uses:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The JOBNAME ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The JOBNAME ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- JOBNAME (OWN) authority to ADD or REMOVE ownership of JOBNAME resources from ACIDs
- JOBNAME(XAUTH) authority to permit or revoke access to JOBNAME resources

The administrator can use any of the following methods to control access to JOBNAME resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: JOBNAME resource class

This example protects the resource by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) JOBNAME(PROD)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) JOBNAME(PROD)
```

This example permits users in the Technical Services Department to access PROD on Fridays only:

```
TSS PERMIT(TECHPROF) JOBNAME(PROD) DAYS(FRIDAY)
```

This example permit users in the Technical Services Department to access PROD:

```
TSS PERMIT(TECHPROF) JOBNAME(PROD)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) JOBNAME(PROD)
```

LDAP Resource Class—Secure LDAP Administration

Valid on z/OS.

Use LDAP to secure authorization roles for LDAP administration.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) LDAP(server)
```

Prefix length

2-26 characters

Capacity of list

1-8 prefixes per TSS command

When used with TSS PERMIT or REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) LDAP(server)
```

Prefix length

1-246 characters

Capacity of list

1-8 prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The LDAP ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADDTO or REMOVE
- The LDAP ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT or REVOKE
- LDAP (OWN) authority to ADDTO or REMOVE ownership of LDAP resources from ACIDs
- LDAP(XAUTH) authority to permit or revoke access to LDAP resources

The administrator can use any of the following methods to control access to SERVER resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: LDAP resource class

This example protects the resource by assigning ownership to the Corporate Department. After ownership is assigned, the administrator can PERMIT access to users or profiles.

```
TSS ADDTO(CORPORAT) LDAP(SERVER1)
```

This example PERMITs a user to access the password administrative role:

```
TSS PERMIT(JOHN01) LDAP(GLDSEC.ADMINROLE.PASSWD) ACCESS(READ)
```

This example revokes access to the password administrative role:

```
TSS REVOKE(JOHN01) LDAP(GLDSEC.ADMINROLE.PASSWD)
```

LFSCLASS Resource Class—Secure MLF LFSCLASS

Valid on z/OS.

Used LFSCLASS to secure LFSCLASS for MLF and other Products.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) LFSCLASS(`*`class`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) LFSCLASS(`*`class`*`)`

Prefix length

One to twenty-six characters

Capacity of list

One to eight prefixes per command

This keyword uses:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The LFSCLASS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The LFSCLASS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- LFSCLASS (OWN) authority to ADD or REMOVE ownership of LFSCLASS resources from ACIDs
- LFSCLASS(XAUTH) authority to permit or revoke access to LFSCLASS resources

The administrator can use any of the following methods to control access to LFSCLASS resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: LFSCLASS resource class

This example protects the resource by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) LFSCLASS(COMMANDS)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) LFSCLASS(COMMANDS)
```

This example permits users in the Technical Services Department to access COMMANDS.XXXXXX on Fridays only:

```
TSS PERMIT(TECHPROF) LFSCLASS(COMMANDS.XXXXXX) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access (COMMANDS.XXXXXX):

```
TSS PERMIT(TECHPROF) LFSCLASS((COMMANDS.XXXXXX)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) LFSCLASS(COMMANDS.XXXXXX)
```

LOGSTRM Resource Class—Secure System Logger

Valid on z/OS

Used LOGSTRM to secure the system logger CICS resources.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) LOGSTRM(oper,oper,oper,...)
```

Prefix length

One to eight characters

Capacity of list

One to five LOGSTRM prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) LOGSTRM(prefix(es))
                 ACCESS(access levels)
```

Prefix length

One to forty-four characters

Capacity of list

One to five prefixes per TSS command

This keyword uses:

- The commands ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The LOGSTRM ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The LOGSTRM ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- LOGSTRM(OWN) authority to ADD or REMOVE ownership of LOGSTRMs from ACIDs
- LOGSTRM(XAUTH) authority to permit or revoke access to LOGSTRMs

The administrator can:

- Specify the access levels: CONTROL, ALL, NONE, READ, UPDATE, ALTER. If ACCESS is not specified, CA Top Secret defaults to READ access
- The administrator can use any of the following methods to control access to LOGSTRMs: Expiration, Facility, Time/Day, and Actions.

MGMTCLAS Resource Class—Secure Management Classes

Valid on z/OS.

Used MGMTCLAS to secure SMS management classes defined by the storage administrator.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) MGMTCLAS(`*`management class`*`,...)`

Prefix length

One to eight characters

Capacity of list

Five management classes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) MGMTCLAS(`*`management class`*`es,...)`

Prefix length

One to eight characters

Capacity of list

One to five management classes per TSS command

This keyword uses:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The MGMTCLAS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The MGMTCLAS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- Update authority to protect user access to specific management classes
- MGMTCLAS(XAUTH) authority to permit or revoke access to management classes that are owned

The administrator can use any of the following methods to control access to MGMTCLAS: Expiration, Facility, Time/Day, and Actions.

Examples: MGMTCLAS resource class

This example protects a fictitious management class of PRODMGMT:

```
TSS ADDTO(DEPT02) MGMTCLAS(PRODMGMT)
```

This example removes ownership:

```
TSS REMOVE(DEPT02) MGMTCLAS(PRODMGMT)
```

This example allows an ACID to use the PRODMGMT management class:

```
TSS PERMIT(DEPT02) MGMTCLAS(PRODMGMT)
```

This example revokes access:

```
TSS REMOVE(DEPT02) MGMTCLAS(PRODMGMT)
```

MQADMIN Resource Class—Secure MQM Commands

Valid on z/OS.

Use MQADMIN to control access to other MQSERIES (MQM) commands.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) MQADMIN(csq1.)
```

Prefix length

Two to twenty-six characters

Capacity of list

One to five MQSERIES resources or prefixes per command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) MQADMIN(csq1)
                 ACCESS(NONE)
```

Prefix length

Two to forty-four characters per prefix

Full name

Two to sixty-two characters

Capacity of list

One to five resources or prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The MQADMIN ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The MQADMIN ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- MQADMIN(OWN) authority

The administrator can:

- Use any of the following methods to control MQSERIES resources: Expiration, Facility, Time/Day, and Actions.
- Specify the accesses: NONE, ALTER, READ, UPDATE, CONTROL, and ALL.

Masking

MQSERIES resources support masking.

Examples: MQADMIN resource class

This example gives the Payroll Department (PAYDEPT) ownership of the MQADMIN resources:

```
TSS ADDTO(PAYDEPT) MQADMIN(CSQ1.)
```

This example removes ownership of these resources:

```
TSS REMOVE(PAYDEPT) MQADMIN(CSQ1.)
```

This example requires full MQSERIES checking of USRMARE:

```
TSS PERMIT(USRMARE) MQADMIN(CSQ1.RESLEVEL)
                    ACCESS(NONE)
```

This example revokes this PERMIT:

```
TSS REVOKE(USRMARE) MQADMIN(CSQ1.)
```

MQCMDS Resource Class—Secure MQM Commands

Valid on z/OS.

Use MQCMDS to secure MQSERIES (MQM) commands.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) MQCMDS(csq1.)
```

Prefix length

Two to twenty-six characters

Capacity of list

One to five MQM resources or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) MQCMDS(csq1.)
                 ACCESS(ALL)
```

Prefix length

Two to twenty-two characters per prefix

Full name

Two to twenty-two characters

Capacity of list

One to five resources or prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The MQCMDS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The MQCMDS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- MQCMDS(OWN) authority

The administrator can:

- Specify the accesses: NONE, ALTER, READ, UPDATE, CONTROL, and ALL.
- Use any of the following methods to control MQSERIES resources: Expiration, Facility, Time/Day, and Actions.

Masking

MQSERIES resources support masking.

Examples: MQCMDS resource class

This example gives the Payroll Department (PAYDEPT) ownership of the MQCMDS resources:

```
TSS ADDTO(PAYDEPT) MQCMDS(CSQ1.)
```

This example removes ownership of the MQCMDS resources:

```
TSS REMOVE(MQQ) MQCMDS(CSQ1.)
```

This example authorizes USRMARE to all MQCMDS resources:

```
TSS PERMIT(USRMARE) MQCMDS(CSQ1.*) ACCESS(ALL)
```

This example revokes USRMARE's authority to MQCMDS resources:

```
TSS REVOKE(USRMARE) MQCMDS(CSQ1.*)
```

MQCONN Resource Class—Secure MQM Connections

Valid on z/OS.

Used MQCONN to secure connections to the MQSERIES (MQM) subsystem.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) MQCONN(csq1.)
```

Prefix length

Two to twenty-six characters

Capacity of list

One to five MQSERIES resources or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) MQCONN(csq1.)
                 ACCESS(level)
```

Prefix length

Two to ten characters per prefix

Full name

Two to ten characters

Capacity of list

One to five resources or prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The MQCONN ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The MQCONN ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- MQCONN(OWN) authority

The administrator can:

- Specify the accesses: NONE, ALTER, READ, UPDATE, CONTROL, and ALL.
- Use any of the following methods to control MQSERIES resources: Expiration, Facility, Time/Day, and Actions.

Masking

MQSERIES resources support masking.

Examples: MQCONN resource class

This example gives the Payroll Department (PAYDEPT) ownership of the MQCONN resources:

```
TSS ADDTO(PAYDEPT) MQCONN(CSQ1.)
```

Ownership of the MQCONN resources is removed:

```
TSS REMOVE(PAYDEPT) MQCONN(CSQ1.)
```

This example authorizes USRMARE to all MQCONN resources:

```
TSS PERMIT(USRMARE) MQCONN(CSQ1.*) ACCESS(ALL)
```

This example revokes USRMARE's authority to MQCONN resources:

```
TSS REVOKE(USRMARE) MQCONN(CSQ1.*)
```

MQNLIST Resource Class—Secure MQM Name Lists

Valid on z/OS.

Use MQNLIST to secure MQSERIES (MQM) name lists.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) MQNLIST(csq1.)
```

Prefix length

Two to twenty-six characters

Capacity of list

One to five MQSERIES resources or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) MQNLIST(csq1.)
                 ACCESS(level)
```

Prefix length

Two to forty-four characters per prefix

Full name

Two to fifty-three characters

Capacity of list

One to five resources or prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The MQNLIST ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The MQNLIST ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- MQNLIST(OWN) authority

The administrator can:

- Use any of the following methods to control MQSERIES resources: Expiration, Facility, Time/Day, and Actions.
- Specify the accesses: NONE, ALTER, READ, UPDATE, CONTROL, and ALL..

Masking

MQSERIES resources support masking.

Examples: MQNLIST resource class

This example gives the Payroll Department (PAYDEPT) ownership of the MQNLIST resources:

```
TSS ADDTO(PAYDEPT) MQNLIST(CSQ1.)
```

Ownership of the MQNLIST resources is removed:

```
TSS REMOVE(PAYDEPT) MQNLIST(CSQ1.)
```

This example authorizes USRMARE to all MQNLIST resources:

```
TSS PERMIT(USRMARE) MQNLIST(CSQ1.*)
                    ACCESS(ALL)
```

This example revokes USRMARE's authority to MQNLIST resources:

```
TSS REVOKE(USRMARE) MQNLIST(CSQ1.*)
```

MQPROC Resource Class—Secure MQM Processes

Valid on z/OS.

Use MQPROC to secure access to MQSERIES (MQM) processes.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) MQPROC(csq1.)
```

Prefix length

Two to twenty-six characters

Capacity of list

One to five MQSERIES resources or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) MQPROC(csq1.*)
                 ACCESS(level)
```

Prefix length

Two to forty-four characters per prefix

Full name

Two to fifty-three characters

Capacity of list

One to five resources or prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The MQPROC ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The MQPROC ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- MQPROC(OWN) authority

The administrator can:

- Specify the accesses: NONE, ALTER, READ, UPDATE, CONTROL, and ALL.
- The administrator can use any of the following methods to control MQSERIES resources: Expiration, Facility, Time/Day, and Actions.

Masking

MQSERIES resources support masking.

Example: MQPROC resource class

This example gives the Payroll Department (PAYDEPT) ownership of MQPROC resources:

```
TSS ADDTO(PAYDEPT) MQPROC(CSQ1.)
```

Ownership of the MQPROC resources is removed:

```
TSS REMOVE(MQQ) MQPROC(CSQ1.)
```

This example authorize USRMARE to all MQPROC resources:

```
TSS PERMIT(USRMARE) MQPROC(CSQ1.*)
                    ACCESS(ALL)
```

This example revokes USRMARE's authority to MQPROC resources:

```
TSS REVOKE(USRMARE) MQPROC(CSQ1.*)
```

MQQUEUE Resource Class—Secure the MQM Queue

Valid on z/OS.

Use MQQUEUE to secure access to the MQSERIES (MQM) queue.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) MQQUEUE(csq1.)
```

Prefix length

Two to twenty-six characters

Capacity of list

One to five MQSERIES resources or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) MQQUEUE(csq1.)
                 ACCESS(level)
```

Prefix length

Two to forty-four characters per prefix

Full name

Two to fifty-three characters

Capacity of list

One to five resources or prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The MQQUEUE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The MQQUEUE ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- MQQUEUE(OWN) authority

The administrator can use:

- Specify the accesses: NONE, ALTER, READ, UPDATE, CONTROL, and ALL.
- Use any of the following methods to control MQSERIES resources: Expiration, Facility, Time/Day, and Actions.

Masking

MQSERIES resources support masking.

Examples: MQQUEUE resource class

This example gives the Payroll Department (PAYDEPT) ownership of the MQQUEUE resources:

```
TSS ADDTO(PAYDEPT) MQQUEUE(CSQ1.)
```

This example removes ownership of the MQQUEUE resources:

```
TSS REMOVE(PAYDEPT) MQQUEUE(CSQ1.)
```

This example authorizes USRMARE to all MQQUEUE resources:

```
TSS PERMIT(USRMARE) MQQUEUE(CSQ1.*)
                    ACCESS(ALL)
```

This example revokes USRMARE's authority to MQQUEUE resources:

```
TSS REVOKE(USRMARE) MQQUEUE(CSQ1.*)
```

MXADMIN Resource Class—Secure MXM Commands

Valid on z/OS.

Use MXADMIN to control access to other MXSERIES (MXM) commands when under the control of WebSphere.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) MXADMIN(csq1.)
```

Prefix length

2 to 26 characters

Capacity of list

1 to 5 MXSERIES resources or prefixes per command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) MXADMIN(csq1)
                 ACCESS(NONE)
```

Full name

2 to 62 characters

Capacity of list

1 to 5 resources or prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The MXADMIN ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The MXADMIN ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- MXADMIN(OWN) authority

The administrator can:

- Use any of the following methods to control MXSERIES resources: Expiration, Facility, Time/Day, and Actions.
- Specify the accesses: NONE, ALTER, READ, UPDATE, CONTROL, and ALL.
- Enable support for mixed-case resource names by entering the following command:

  ```
  TSS REP(MXADMIN) ATTR(MIXED)
  ```

Masking

MXSERIES resources support masking.

Examples: MXADMIN resource class

This example gives the Payroll Department (PAYDEPT) ownership of the MXADMIN resources:

```
TSS ADDTO(PAYDEPT) MXADMIN(CSQ1.)
```

This example removes ownership of these resources:

```
TSS REMOVE(PAYDEPT) MXADMIN(CSQ1.)
```

This example requires full MXSERIES checking of USRMARE:

```
TSS PERMIT(USRMARE) MXADMIN(CSQ1.RESLEVEL)
                    ACCESS(NONE)
```

This example revokes this PERMIT:

```
TSS REVOKE(USRMARE) MXADMIN(CSQ1.)
```

MXNLIST Resource Class—Secure MXM Name Lists

Valid on z/OS.

Use MXNLIST to secure MXSERIES (MXM) name lists when under the control of WebSphere.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) MXNLIST(csq1.)
```

Prefix length

2 to 26 characters

Capacity of list

1 to 5 MXSERIES resources or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) MXNLIST(csq1.)
                 ACCESS(level)
```

Full name

2 to 53 characters

Capacity of list

1 to 5 resources or prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The MXNLIST ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The MXNLIST ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- MXNLIST(OWN) authority

The administrator can:

- Use any of the following methods to control MXSERIES resources: Expiration, Facility, Time/Day, and Actions
- Specify the accesses: NONE, ALTER, READ, UPDATE, CONTROL, and ALL.
- Enable support for mixed-case resource names by entering the following command:

  ```
  TSS REP(MXNLIST) ATTR(MIXED)
  ```

Masking

MXSERIES resources support masking.

Examples: MXNLIST resource class

This example gives the Payroll Department (PAYDEPT) ownership of the MXNLIST resources:

```
TSS ADDTO(PAYDEPT) MXNLIST(CSQ1.)
```

This example removes ownership of the MXNLIST resources:

```
TSS REMOVE(PAYDEPT) MXNLIST(CSQ1.)
```

This example authorizes USRMARE to all MXNLIST resources:

```
TSS PERMIT(USRMARE) MXNLIST(CSQ1.*)
                    ACCESS(ALL)
```

This example revokes USRMARE's authority to MXNLIST resources:

```
TSS REVOKE(USRMARE) MXNLIST(CSQ1.*)
```

MXPROC Resource Class—Secure MXM Topics

Valid on z/OS.

Use MXPROC to secure access to MXSERIES (MXM) processes when under the control of WebSphere.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) MXPROC(csq1.)
```

Prefix length

2 to 26 characters

Capacity of list

1 to 5 MXSERIES resources or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) MXPROC(csq1.*)
                 ACCESS(level)
```

Full name

2 to 53 characters

Capacity of list

1 to 5 resources or prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The MXPROC ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The MXPROC ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- MXPROC(OWN) authority

The administrator can:

- Specify the accesses: NONE, ALTER, READ, UPDATE, CONTROL, and ALL.
- The administrator can use any of the following methods to control MXSERIES resources: Expiration, Facility, Time/Day, and Actions.
- Enable support for mixed-case resource names by entering the following command:

  ```
  TSS REP(MXPROC) ATTR(MIXED)
  ```

Masking

MXSERIES resources support masking.

Example: MXPROC resource class

This example gives the Payroll Department (PAYDEPT) ownership of MXPROC resources:

```
TSS ADDTO(PAYDEPT) MXPROC(CSQ1.)
```

This example removes ownership of the MXPROC resources:

```
TSS REMOVE(MXM) MXPROC(CSQ1.)
```

This example authorize USRMARE to all MXPROC resources:

```
TSS PERMIT(USRMARE) MXPROC(CSQ1.*)
                    ACCESS(ALL)
```

This example revokes USRMARE's authority to MXPROC resources:

```
TSS REVOKE(USRMARE) MXPROC(CSQ1.*)
```

MXQUEUE Resource Class—Secure the MXM Queue

Valid on z/OS.

Use MXQUEUE to secure access to the MXSERIES (MXM) queue when under the control of WebSphere.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) MXQUEUE(csq1.)
```

Prefix length

2 to 26 characters

Capacity of list

1 to 5 MXSERIES resources or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) MXQUEUE(csq1.)
                 ACCESS(level)
```

Full name

2 to 53 characters

Capacity of list

1 to 5 resources or prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The MXQUEUE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The MXQUEUE ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- MXQUEUE(OWN) authority

The administrator can use:

- Specify the accesses: NONE, ALTER, READ, UPDATE, CONTROL, and ALL.
- Use any of the following methods to control MXSERIES resources: Expiration, Facility, Time/Day, and Actions.
- Enable support for mixed-case resource names by entering the following command:

  ```
  TSS REP(MXQUEUE) ATTR(MIXED)
  ```

Masking

MXSERIES resources support masking.

Examples: MXQUEUE resource class

This example gives the Payroll Department (PAYDEPT) ownership of the MXQUEUE resources:

```
TSS ADDTO(PAYDEPT) MXQUEUE(CSQ1.)
```

This example removes ownership of the MXQUEUE resources:

```
TSS REMOVE(PAYDEPT) MXQUEUE(CSQ1.)
```

This example authorizes USRMARE to all MXQUEUE resources:

```
TSS PERMIT(USRMARE) MXQUEUE(CSQ1.*)
                    ACCESS(ALL)
```

This example revokes USRMARE's authority to MXQUEUE resources:

```
TSS REVOKE(USRMARE) MXQUEUE(CSQ1.*)
```

MXTOPIC Resource Class—Secure MXM Topics

Valid on z/OS.

Use MXTOPIC to secure access to MXSERIES (MXM) topics when under the control of WebSphere.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) MXPTOPICcsq1.)
```

Prefix length

2 to 26 characters

Capacity of list

1 to 5 MXSERIES resources or prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) MXTOPIC(csq1.*)
                 ACCESS(level)
```

Full name

2 to 246 characters

Capacity of list

1 to 5 resources or prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, WHOOWNS, and WHOHAS
- The MXPROC ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The MXPROC ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- MXPROC(OWN) authority

The administrator can:

- Specify the accesses: NONE, ALTER, READ, UPDATE, CONTROL, and ALL.
- The administrator can use any of the following methods to control MXSERIES resources: Expiration, Facility, Time/Day, and Actions.
- Enable support for mixed-case resource names by entering the following command:

  ```
  TSS REP(MXTOPIC) ATTR(MIXED)
  ```

Masking

MXSERIES resources support masking.

Example: MXTOPIC resource class

This example gives the Payroll Department (PAYDEPT) ownership of MXTOPIC resources:

```
TSS ADDTO(PAYDEPT) MXTOPIC(CSQ1.)
```

This example removes ownership of the MXTOPIC resources:

```
TSS REMOVE(MXM) MXPTOPIC(CSQ1.)
```

This example authorize USRMARE to all MXTOPIC resources:

```
TSS PERMIT(USRMARE) MXTOPIC(CSQ1.*)
                    ACCESS(ALL)
```

This example revokes USRMARE's authority to MXTOPIC resources:

```
TSS REVOKE(USRMARE) MXTOPIC(CSQ1.*)
```

NETCMDS Resource Class—Secure NETVIEW NETCMDS

Valid on z/OS.

Use NETCMDS to secure NETCMDS for NETVIEW.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) NETCMDS(`*`commands`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) NETCMDS(`*`commands`*`)`

Prefix length

One to forty-four characters

Capacity of list

One to eight prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The NETCMDS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The NETCMDS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- NETCMDS (OWN) authority to ADD or REMOVE ownership of NETCMDS resources from ACIDs
- NETCMDS(XAUTH) authority to permit or revoke access to NETCMDS resources

The administrator can use any of the following methods to control access to NETCMDS resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: NETCMDS resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) NETCMDS(COMMANDS)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) NETCMDS(COMMANDS)
```

This example permit users in the Technical Services Department to access COMMANDS.XXXXXX on Fridays only:

```
TSS PERMIT(TECHPROF) NETCMDS(COMMANDS.XXXXXX)
                     DAYS(FRIDAY)
```

This example permit users in the Technical Services Department to access (COMMANDS.XXXXXX):

```
TSS PERMIT(TECHPROF) NETCMDS((COMMANDS.XXXXXX)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) NETCMDS(COMMANDS.XXXXXX)
```

NETSPAN Resource Class—Secure NETVIEW NETSPAN

Valid on z/OS.

Used NETSPAN to secure NETSPAN for NETVIEW.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) NETSPAN(`*`commands`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) NETSPAN(`*`commands`*`)`

Prefix length

One to forty-four characters

Capacity of list

One to eight prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The NETSPAN ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The NETSPAN ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- NETSPAN (OWN) authority to ADD or REMOVE ownership of NETSPAN resources from ACIDs
- NETSPAN(XAUTH) authority to permit or revoke access to NETSPAN resources

The administrator can use any of the following methods to control access to NETSPAN resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: NETSPAN resource class

This example protects the resource by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) NETSPAN(COMMANDS)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) NETSPAN(COMMANDS)
```

This example permits users in the Technical Services Department to access COMMANDS.XXXXXX on Fridays only:

```
TSS PERMIT(TECHPROF) NETSPAN(COMMANDS.XXXXXX) DAYS(FRIDAY)
```

This example permit users in the Technical Services Department to access (COMMANDS.XXXXXX):

```
TSS PERMIT(TECHPROF) NETSPAN((COMMANDS.XXXXXX)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) NETSPAN(COMMANDS.XXXXXX)
```

NODES Resource Class— Control NJE Jobs and SYSOUT

Valid on z/OS and z/VM.

Use NODES to control incoming NJE jobs and SYSOUT, and optionally, to assign a userid as the owner of the job or SYSOUT on this system.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) NODES(node,...)
```

Prefix length

Two to twenty-six characters

Capacity of list

One to five node names or prefixes per TSS command

Generic prefixing allows the administrator to group a set of similar nodes together, and define them by one generic prefix.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) NODES(node(s))
                  ACCESS(levels)
                 [NJEACID(acid)]
```

Prefix length

Two to forty-four characters

Capacity of list

One to five nodes per TSS command

Note: A fully qualified node is PERMITted to an ACID by enclosing in single quotation marks. This will indicate that it is defined to CA Top Secret, not as a prefix, but by its fully qualified name.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The NODES ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The NODES ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- NODES(OWN) authority to ADD or REMOVE ownership of nodes from ACIDs

The administrator can:

- Use any of the following methods to control access to data sets: Expiration, Facility, Program Pathing, Time/Day, and Actions.
- Specify the access levels: CONTROL, NONE, READ, and UPDATE. If ACCESS is not specified, CA Top Secret defaults to ALL access.

Masking

Node masking is another method of reducing the number of node definitions to implement widespread node protection.

Examples: NODES resource class

This example gives USER01 ownership of jobs submitted from ALPHA2.USERJ:

```
TSS ADDTO(USER01) NODES(ALPHA.USERJ*)
```

This example removes ownership:

```
TSS REMOVE(USER01) NODES(ALPHA.USERJ*)
```

If node ALPHA is not allowed to execute jobs on node ALPHA.USERJ:

```
TSS PERMIT(ALL) NODES(ALPHA.USERJ.) ACCESS(NONE)
```

If an exception is made for userid X123:

```
TSS PERMIT(ALL) NODES(ALPHA.USERJ.X123) ACCESS(UPDATE)
```

If any job submitted from node BETA is to run without any password checking:

```
TSS PERMIT(ALL) NODES(BETA.USERJ) ACCESS(CONTROL)
```

OPCMD Resource Class—Secure SVC 34 Commands

Valid on z/OS.

Use OPCMD to secure system operator commands issued by programs using SVC 34. For example, SDSF, TSO, OPER, or other installation-written programs.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) OPCMD(`*`prefix(es)`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) OPCMD(`*`prefix(es)`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The OPCMD ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The OPCMD ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- OPCMD(OWN) authority to ADD or REMOVE ownership of OPCMDs from ACIDs
- OPCMD(XAUTH) authority to permit or revoke the use of OPCMDs

The administrator can use any of the following methods to control access to the OPCMD resource: Expiration, Facility, Time/Day, and Actions.

- **Note:** In OS/390 3.1.3 and above environments, the OPERCMDS resource class may be used instead.

Example: OPCMD resource class

This example assigns ownership of the DISPLAY console operator command to Department 01:

```
TSS ADDTO(DEPT01) OPCMD(DISPLAY)
```

This example removes ownership:

```
TSS REMOVE(DEPT01) OPCMD(DISPLAY)
```

This example permits a group of technical support users to use the SET OPCMD:

```
TSS PERMIT(TECHPRO) OPCMD(SET)
```

This example revokes access:

```
TSS REVOKE(TECHPRO) OPCMD(SET)
```

OPERCMDS Resource Class—Secure JES Operator Commands

Valid on z/OS.

Use OPERCMDS to secure JES, z/OS operator commands.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) OPERCMDS(`*`oper`*`,`*`oper`*`...)`

Prefix length

Two to twenty-six characters

Capacity of list

One to eight prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) OPERCMDS(`*`prefix(es)`*`)`

Prefix length

Two to forty-four characters.

Capacity of list

One to eight prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The OPERCMDS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The OPERCMDS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- OPERCMDS(OWN) authority to ADD or REMOVE ownership of OPERCMDS from ACIDs
- OPERCMDS(XAUTH) authority to permit or revoke access OPERCMDS

The administrator can:

- Specify the access levels: ALL, NONE, READ, UPDATE, and CONTROL**.** If ACCESS is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to OPERCMDS: Expiration, Facility, Time/Day, and Actions.
- **Note:** Only z/OS, ESA release 3.1.3 and above offer support for all resources.

Masking

The OPERCMDS resource class supports all masking characters.

Examples: OPERCMDS resource class

This example assigns ownership to USER01, so that ACID can cancel a TSO session:

```
TSS ADDTO(USER01) OPERCMDS(ABC.ABC.ABC)
```

This example removes ownership:

```
TSS REMOVE(USER01) OPERCMDS(ABC.ABC.ABC)
```

This example allows any user to cancel their own TSO session (assuming that the user has access to a subsystem console):

```
TSS PERMIT(ALL) OPERCMDS(ABC.ABC.ABC.%) ACCESS(UPDATE)
```

This example revokes access:

```
TSS REVOKE(ALL) OPERCMDS(ABC.ABC.ABC.%)
```

OTRAN Resource Class—Secure Ownable Transactions

Valid on z/OS and z/VSE.

Use OTRAN to secure ownable transactions for CICS, IMS, and CA-IDMS that are protected by OTRAN. OTRAN is fully supported as a NONMASK resource (set by default). OTRAN in the IMS environment does not support the MASK RDT attribute.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*acid*`) OTRAN(`*transaction-id*`)`

Prefix length

One to eight characters

Capacity of list

One to five transaction per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`) OTRAN(`*transaction-id*`)`

Prefix length

One to eight characters

Capacity of list

One to five transactions per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- OTRAN ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- OTRAN ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- OTRAN(OWN) authority to ADD or REMOVE OTRAN from ACIDs
- OTRAN(XAUTH) authority to permit or revoke access to transactions

The administrator can:

- Specify the access levels: INQUIRE, SET, EXECUTE, ALL, UPDATE, INSTALL, and COLLECT. If ACCESS is not specified, CA Top Secret defaults to EXECUTE access.
- Use any of the following methods to control access to the OTRAN resource: Expiration, Facility, Time/Day, and Actions.

Examples: OTRAN resource class

This example protects the CICS transaction, PAYR, by assigning ownership to the Payroll Department (PAYDEPT):

```
TSS ADDTO(PAYDEPT) OTRAN(PAYR)
```

This example removes ownership:

```
TSS REMOVE(PAYDEPT) OTRAN(PAYR)
```

This example permits an ACID, PAYPROG, to access the transaction PA01 through CICS only:

```
TSS PERMIT(PAYPROG) OTRAN(PA01)
                    FACILITY(CICS)
```

This example grants USER01 execute, collect, and set access to the TRANS1 ownable transaction:

```
TSS PERMIT(USER01) OTRAN(TRANS1) ACCESS(EXECUTE,COLLECT,SET)
```

This example revokes access:

```
TSS REVOKE(PAYPROG) OTRAN(PA01)
```

Transactions that require additional security is defined to require the signon password entered with each use. This is an attempt to prevent certain sensitive transactions from being entered by an unauthorized individual at an unlocked terminal. Once the resource is owned, password re-verification is indicated on the PERMIT by the ACTION (REVERIFY) parameter as shown next:

```
TSS PERMIT(SUPR01) OTRAN(PROD)
                   ACTION(REVERIFY)
```

PANAPT Resource Class—Secure PANAPT for PANVELT

Valid on z/OS.

Use PANAPT to secures PANAPT for PANVELT.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) PANAPT(`*`commands`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) PANAPT(`*`commands`*`)`

Prefix length

One to forty-four characters

Capacity of list

One to eight prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The PANAPT ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The PANAPT ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- PANAPT (OWN) authority to ADD or REMOVE ownership of PANAPT resources from ACIDs
- PANAPT(XAUTH) authority to permit or revoke access to PANAPT resources

The administrator can use any of the following methods to control access to PANAPT resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Example: PANAPT resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) PANAPT(COMMANDS)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) PANAPT(COMMANDS)
```

This example permits users in the Technical Services Department to access COMMANDS.XXXXXX on Fridays only:

```
TSS PERMIT(TECHPROF) PANAPT(COMMANDS.XXXXXX)
                     DAYS(FRIDAY)
```

This example permit users in the Technical Services Department to access (COMMANDS.XXXXXX):

```
TSS PERMIT(TECHPROF) PANAPT((COMMANDS.XXXXXX)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) PANAPT(COMMANDS.XXXXXX)
```

PANEL Resource Class—Secure Panels

Valid on z/OS and z/VM.

Use PANEL to secure authority to ISPF, CICS and REXX panels.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) PANEL(`*`prefix(es)`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) PANEL(`*`prefix(es)`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The PANEL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The PANEL ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- PANEL(OWN) authority to ADD or REMOVE authority to control access to CA product specific panels
- PANEL(XAUTH) authority to permit or revoke the authority to CA product specific panels

The administrator can:

- Specify the access levels: ALL, NONE, READ, WRITE, UPDATE, CONTROL, and SCRATCH. If ACCESS is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to PANEL resources: Expiration, Facility, Time/Day, and Actions.

Examples: PANEL resource class

This example gives PRDCNTL ownership of the BrightStor CA-1 panel named LOABCDE:

```
TSS ADDTO(PRDCNTL) PANEL(LOABCDE)
```

This example removes ownership:

```
TSS REMOVE(PRDCNTL) PANEL(LOABCDE)
```

This example permits ACID, TAPELIB, the ability to use the READ functions of BrightStor CA-1 panel, LOABCDE:

```
TSS PERMIT(TAPELIB) PANEL(LOABCDE) ACCESS(READ)
```

This example revokes access:

```
TSS REVOKE(TAPELIB) PANEL(LOABCDE) ACCESS(NONE)
```

PDSMEMn Resource Class—Secure Partitioned Data Set Members

Valid on z/OS.

Use PDSMEM*n* to secure an individual or group of Partitioned Data Set members. (where *n* is 1 to 8)

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) PDSMEMn(member|<DIR>|<BYPASS>)
```

Prefix length

One to eight characters

Capacity of list

One to five PDS members per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) PDSMEMn(prefixes)
                 ACCESS(access-level(s))
```

Prefix length

One to eight characters

Capacity of list

One to five PDS members per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, and REVOKE
- The PDSMEM*n* ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The PDSMEMn ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- PDSMEMn(OWN) authority to ADD or REMOVE PDS members from ACIDs
- PDSMEM*n*(XAUTH) authority to permit or revoke PDS members

The administrator can:

- Specify the access levels: NONE, ALTER, READ, UPDATE, and ALL.
- The administrator can use any of the following methods to control access to PDS members: Expiration, Facility, Time/Day, and Privpgm.

Masking

Masking is not allowed; however, ownership ior permission is granted by generic prefixing within resource class.

Example: PDSMEMn resource class

This example assigns ownership of a PDS member-name family, prefixed by IEA:

```
TSS ADDTO(SYSPROG) PDSMEM(IEA)
```

This example removes ownership:

```
TSS REMOVE(SYSPROG) PDSMEM(IEA)
```

This example permits access for READ and UPDATE to member IEASYS00:

```
TSS PERMIT(SYSPROG) PDSMEM(IEASY00)
                    ACCESS(UPDATE)
```

READ is implied by UPDATE.

This example revokes access:

```
TSS REVOKE(SYSPROG) PDSMEM(IEASY00)
```

PKISERV Resource Class—Control Access to R_Pkiserv Administrative Functions

Valid on z/OS.

Use PKISERV to control access to R_PKIServ administrative functions on a more granular level than with IRR.RPKISERV.PKIADMIN resources in the IBMFAC resource class.

Note: For information on setting the permissions, see IBM documentation *z/OS Cryptographic Services PKI Services Service Guide and Reference*.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) PKISERV(MYDOMAIN)
```

Prefix length

2 - 26 characters.

Capacity of list

1 - 5 prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) PKISERV(MYDOMAIN)
```

Resource length

1 - 246 characters.

This resource class is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS.
- The ACID types User, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE.
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE.
- PKISERV(OWN) authority to ADD or REMOVE ownership of PKISERV resources from ACIDs.
- PKISERV(XAUTH) authority to PERMIT or REVOKE access to PKISERV resources.

The administrator can use any of the following methods to control access to PKISERV resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: PKISERV resource class

- This example protects the resource by assigning ownership to the Corporate Department ACID:

  ```
  TSS ADDTO(CORPORAT) PKISERV(MYDOMAIN)
  ```

  The administrator can now PERMIT access to users or profiles that require access.
- This example removes ownership:

  ```
  TSS REMOVE(CORPORAT) PKISERV(MYDOMAIN)
  ```
- This example permits users to access PKISERV(MYDOMAIN.QUERYREQS.1BYSSL):

  ```
  TSS PERMIT(TECHUSER) PKISERV(MYDOMAIN.QUERYREQS.1BYSSL)
  ```

PPT Resource Class—Secure the CICS PPT

Valid on z/OS and z/VSE.

Use PPT to secure program entries in the CICS Program Propagation Table (PPT).

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) PPT(`*`program`*`)`

Prefix length

One to eight characters

Capacity of list

One to five CICS PPTs per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) PPT(`*`program`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The PPT ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The PPT ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- PPT(OWN) authority to ADD or REMOVE PPTs from ACIDs
- PPT(XAUTH) authority to permit or revoke access to transactions

The administrator can:

- Specify the access levels: INQUIRE, SET, EXECUTE, ALL, NONE, and UPDATE. If ACCESS is not specified, CA Top Secret defaults to EXECUTE access.
- Use any of the following methods to control access to PPTs: Expiration, Facility, Time/Day, and Actions.

Example: PPT resource class

This example assigns ownership of a PPT name, CP12388, to a department:

```
TSS ADDTO(CICSCORP) PPT(CP12388)
```

This example removes ownership:

```
TSS REMOVE(CICSCORP) PPT(CP12388)
```

This example permits a test CICS user access to a program, APPLPROG:

```
TSS PERMIT(GKM75) PPT(APPLPROG)
```

This example revokes access:

```
TSS REVOKE(GKM75) PPT(APPLPROG)
```

This example grants USER01 inquire and set access to the APPLPROG program propagation table:

```
TSS PERMIT(USER01) PPT(APPLPROG) ACCESS(INQUIRE,SET)
```

PROGRAM Resource Class—Secure Programs and Utilities

Valid on z/OS and z/VSE.

Use PROGRAM to secure system programs and utilities.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) PROGRAM(`*`program`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) PROGRAM(`*`program`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes or names per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The PROGRAM ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The PROGRAM ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- PROGRAM(OWN) authority to ADD or REMOVE PROGRAMs from ACIDs
- PROGRAM(XAUTH) authority to permit or revoke access to transactions

The administrator can use any of the following methods to control access to PROGRAMS: Expiration, Facility, Time/Day, and Actions. Sysid, Timerec, and Calendar restrictions are not supported.

Protection

Programs that are protected include: TSO commands, and those specified by 'EXEC PGM=program' in JCL statements; all programs executed internally via CALL, LINK, LOAD, XCTL, ATTACH, EXECUTE and CA-IDMS programs.

Examples: PROGRAM Resource Class

This example protects the use of all system utilities, IEH5000 and IEASSST, by assigning ownership to the Corporate Department ACID, and subsequently PERMIT restricted access to users or profiles:

```
TSS ADDTO(CORPRAT) PROGRAM(IEH,IEASSST)
```

This example removes ownership:

```
TSS REMOVE(CORPRAT) PROGRAM(IEH,IEASSST)
```

This example permits use of the SUPERZAP program from Batch only:

```
TSS PERMIT(SYSAU2) PROGRAM(IMASPZAP)
                   FACILITY(BATCH)
```

This example revokes access:

```
TSS REVOKE(SYSAU2) PROGRAM(IMASPZAP) REMOVE
```

PROPCNTL Resource Class—Secure Special ACIDs

Valid on z/OS.

Use PROPCNTL to secure special ACIDs that are not subject to automatic propagation of batch jobs.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) PROPCNTL(acid,...)
```

Prefix length

One to eight characters

Capacity of list

five ACIDs per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, ADMIN, DEADMIN,and WHOOWNS
- The PROPCNTL ACID types User, DCA, VCA, ZCA, LSCA, SCA, and MSCA
- Update authority to ADD or REMOVE PROPCNTL from ACIDs

Examples: PROPCNTL resource class

This example prevents the IMS ACID from being automatically propagated:

```
TSS ADDTO(IMSDEPT) PROPCNTL(IMS)
```

This example removes ownership:

```
TSS REMOVE(IMSDEPT) PROPCNTL(IMS)
```

PSB Resource Class—Secure the IMS Program Specification Block

Valid on z/OS.

Use PSB to secures IMS Program Specification Block.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) PSB(`*`oper`*`,`*`oper`*`,...)`

Prefix length

One to eight characters

Capacity of list

One to five PSBs per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) PSB(`*`prefix(es)`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The PSB ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The PSB ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- PSB(OWN) authority to ADD or REMOVE PSBs from ACIDs
- PSB(XAUTH) authority to permit or revoke access to PSB

The administrator can specify the access levels: INQUIRE, SET, ALL, NONE, READ, and UPDATE.

Access levels INQUIRE, SET and UPDATE are for CICS (CEMT) access to the PCBs that are active in a CICS region. Access level READ is the level that authorizes access to the PSB through the control region or standalone batch region. Access level ALL allows access to PSB from both CICS and IMS in any of the above.

If ACCESS is not specified, CA Top Secret defaults to READ access.

The administrator can use any of the following methods to control access to PSBs: Expiration, Facility, Pathing, Time/Day, and Actions.

Examples: PSB resource class

To assign ownership of the PSB, PERSDB, to the Personnel Department:

```
TSS ADDTO(PERDEPT) PSB(PERSDB)
```

This example removes ownership:

```
TSS REMOVE(PERDEPT) PSB(PERSDB)
```

This example permits users to access the department's PSBs:

```
TSS PERMIT(DTG05) PSB(PERSDB)
```

This example revokes access:

```
TSS REVOKE(DTG05) PSB(PERSDB)
```

PSFMPL Resource Class—Secure Output Labeling Suppression

Valid on z/OS.

Use PSFMPL to secure user suppression of output labeling.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) PSFMPL(`*`prefix(es)`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) PSFMPL(`*`prefix(es)`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The PSFMPL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The PSFMPL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- PSFMPL(OWN) authority to ADD or REMOVE authority to control suppression of page labeling under PSF by users
- PSFMPL(XAUTH) authority to permit or revoke user controlled suppression of data page labeling by ACIDs

The administrator can:

- Specify the access levels NONE and READ.
- The administrator can use any of the following methods to control access to PSFMPL resources: Expiration, Facility, Time/Day, and Actions.

Examples: PSFMPL resource class

This example gives PRDCNTL ownership of PSFMPL resources to control suppression of output data page labeling:

```
TSS ADDTO(PRDCNTL) PSFMPL(PSF.)
```

This example removes ownership:

```
TSS REMOVE(PRDCNTL) PSFMPL(PSF.)
```

This example permits ACID JOBOUT which is assigned to a batch job requests suppression of data page labeling for its output:

```
TSS PERMIT(JOBOUT) PSFMPL(PSF.DPAGELBL)
                   ACCESS(READ)
```

This example denies authority for all other users able to request data page labeling suppression:

```
TSS PERMIT(ALL) PSFMPL(PSF.DPAGELBL)
                ACCESS(NONE)
```

PTKTDATA Resource Class—PassTicket Data Resources

Valid on z/OS.

Use the PTKTDATA resource class to define data resources for PassTickets used with R_GenSec and R_TicketServ callable services. These resources are based on the application ID and target used in the PassTicket function.

This resource class has the following format:

```
TSS ADD(RDT) RESCLASS(PTKTDATA)
             ACLST(ALL,READ,UPDATE)
             MAXLEN(37)
```

ACLST

Defines the access permissions for the PTKTDATA resource.

MAXLEN

Defines the maximum length of the PTKTDATA resource name

This keyword is used with:

- The commands ADDTO, REMOVE, PERMIT, and REVOKE
- The RDT Record only
- MISC1(RDT) authority

Examples: PTKDATA resource class

This example indicates that IRRPTAUT is owned:

```
TSS ADD(tssdept) PTKTDATA(IRRPTAUT)
```

This example adds a permit to the component:

```
TSS PER(tsscomp1) PTKTDATA(IRRPTAUTH.aaaaaaaa.uuuuuuuu)
ACCESS(READ,UPDATE)
```

aaaaaaaa

The application.

uuuuuuuu

The user.

RECIPID Resource Class—Secure CADISPATCH RECIPID

Valid on z/OS.

Use RECIPID to secure RECIPID for CA-DISPATCH.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) RECIPID(`*`oper`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) RECIPID(`*`oper`*`)`

Prefix length

One to sixteen characters

Capacity of list

One to eight prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The RECIPID ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The RECIPID ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- RECIPID (OWN) authority to ADD or REMOVE ownership of RECIPID resources from ACIDs
- RECIPID(XAUTH) authority to permit or revoke access to RECIPID resources

The administrator can use any of the following methods to control access to RECIPID resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: RECIPID resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) RECIPID(ABCDEFGH)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) RECIPID(ABCDEFGH)
```

This example permits users in the Technical Services Department to access ABCDEFGH.XXXXXXXX on Fridays only:

```
TSS PERMIT(TECHPROF) RECIPID(ABCDEFGH.XXXXXXXX) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access ABCDEFGH.XXXXXXXX:

```
TSS PERMIT(TECHPROF) RECIPID(ABCDEFGH.XXXXXXXX)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) RECIPID(ABCDEFGH.XXXXXXXX)
```

RESLIST Resource Class—Secure TSSAI RESLIST

Valid on z/OS and z/VSE.

Use RESLIST to secure RESLIST when using TSSAI to do RESLIST calls.

Violations of RESLIST are only acknowledged by a failure of the RESLIST function, with TSSRC=8 (not authorized) returned in the parameter list. No messages are generated and events involving this resource are not logged. For information, see the *User Guide*.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) RESLIST(`*`list`*`)`

Prefix length

One to eight characters.

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) RESLIST(`*`list`*`)`

Prefix length

One to eight characters

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The RESLIST ACID types User, Department, Division, Zone, DCA, VCA, ZCA, SCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The RESLIST ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- RESLIST (OWN) authority to ADD or REMOVE ownership of RESLIST resources from ACIDs
- RESLIST(XAUTH) authority to permit or revoke access to RESLIST resources

The administrator can use any of the following methods to control access to RESLIST resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: RESLIST resource class

This example protects the resource by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) RESLIST(RESLIST1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) RESLIST(RESLIST1)
```

This example revokes access:

```
TSS REVOKE(TECHUSER) RESLIST(RESLIST1)
```

RODMMGR Resource Class—Secure NETVIEW RODMMGR

Valid on z/OS.

Use RODMMGR to secure RODMMGR for NETVIEW.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) RODMMGR(`*`commands`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) RODMMGR(`*`commands`*`)`

Prefix length

One to forty-four characters

Capacity of list

One to eight prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The RODMMGR ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The RODMMGR ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- RODMMGR (OWN) authority to ADD or REMOVE ownership of RODMMGR resources from ACIDs
- RODMMGR(XAUTH) authority to permit or revoke access to RODMMGR resources

The administrator can use any of the following methods to control access to RODMMGR resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: RODMMGR resource class

This example protects the resource by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) RODMMGR(COMMANDS)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) RODMMGR(COMMANDS)
```

This example permits users in the Technical Services Department to access COMMANDS.XXXXXX on Fridays only:

```
TSS PERMIT(TECHPROF) RODMMGR(COMMANDS.XXXXXX)
                     DAYS(FRIDAY)
```

This example permit users in the Technical Services Department to access (COMMANDS.XXXXXX):

```
TSS PERMIT(TECHPROF) RODMMGR((COMMANDS.XXXXXX)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) RODMMGR(COMMANDS.XXXXXX)
```

ROSRES Resource Class—Determine CA Roscoe CA—Vollie Command Ownership

Valid on z/OS and z/VSE.

Use ROSRES to determine who has ownership or control over CA Roscoe/VA—Vollie commands.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*acid*`) ROSRES(`*resource*`,...)`

Prefix length

One to eight characters

Capacity of list

Five ROSRES resources per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`) ROSRES(`*resource*`,...)`

Prefix length

One to 44 characters

Capacity of list

Five ROSRES resources per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ROSRES ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The ROSRES ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, MSCA, and ALL when used with TSS PERMIT/REVOKE
- ROSRES(OWN) authority to protect user access to CA Roscoe ROSRES resources
- ROSRES(XAUTH) authority to permit or revoke access to CA Roscoe resources

The administrator can:

- Use any of the following methods to control access to ROSRES: Expiration, Facility, Time/Day, and Actions.
- Specify the access levels: CREATE, ALL, NONE, READ, UPDATE, and WRITE. If ACCESS is not specified, CA Top Secret defaults to NONE access.

Examples: ROSRES resource class

This example adds a ROSRES of JBS.ROSCMD:

```
TSS ADDTO(DEPT02) ROSRES(JBS.ROSCMD)
```

This example removes ownership:

```
TSS REMOVE(DEPT02) ROSRES(JBS.ROSCMD)
```

This example allows an ACID to use ROSRES:

```
TSS PERMIT(DEPT02) ROSRES([rosid.]ROSCMD.,...)
```

This example revokes access:

```
TSS REVOKE(DEPT02) ROSRES([rosid.]ROSCMD.,...)
```

SCHEDULE Resource Class—Secure SCHEDULE for CA7

Valid on z/OS.SCHEDULE for CA7

Use SCHEDULE to secure SCHEDULE for CA7.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*acid*`) SCHEDULE(`*prod*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`) SCHEDULE(`*prod*`)`

Prefix length

One to eight characters

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The SCHEDULE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The SCHEDULE ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- SCHEDULE (OWN) authority to ADD or REMOVE ownership of SCHEDULE resources from ACIDs
- RECIPID(XAUTH) authority to permit or revoke access to SCHEDULE resources

The administrator can use any of the following methods to control access to SCHEDULE resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: SCHEDULE resource class

This example protects the resourceby assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) SCHEDULE(PROD1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) SCHEDULE(PROD1)
```

This example permits users in the Technical Services Department to access PROD1 on Fridays only:

```
TSS PERMIT(TECHPROF) SCHEDULE(PROD1) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access PROD1

```
TSS PERMIT(TECHPROF) SCHEDULE(PROD1)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) SCHEDULE(PROD1)
```

SDSF Resource Class—Secure SDSF Commands

Valid on z/OS.

Use SDSF to secure SDSF 1.3 commands, functions and other resources.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) SDSF(oper,oper,...)
```

Prefix length

Two to twenty-six characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) SDSF(prefixes(s))
                 ACCESS(access level(s))
```

Prefix length

Two to sixty-three characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The SDSF ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The SDSF ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- SDSF(OWN) authority to ADD or REMOVE SDSF resources
- SDSF(XAUTH) authority to permit or revoke SDSF resources

The administrator can:

- Use any of the following methods to control access to SDSF resources: Expiration, Facility, Time/Day, and Actions.
- Specify any of the following access levels: NONE, READ, UPDATE, CONTROL, and ALL.

Masking

The SDSF resource class supports all masking characters.

Example: SDSF Resource Class—Secure

This example defines DEPT01 the ability of users to issue z/OS and JES2 commands on the SDSF command line via the "/" command:

```
TSS ADDTO(DEPT01) SDSF(ISFOPER.SYSTEM)
```

The administrator may remove the ability:

```
TSS REMOVE(DEPT01) SDSF(ISFOPER.SYSTEM)
```

This example denies USER42 the ability to use SDSF's PR (printer) and INIT (initiator) displays:

```
TSS PERMIT(USER42) SDSF(ISFCMD.ODSP.INITIATOR.*/ISFCMD.ODSP.PRINTER.*)
                   ACCESS(NONE)
```

SERVAUTH Resource Class—Secure TCP/IP Resources

Valid on z/OS.

Use SERVAUTH to protect TCP/IP resources from unauthorized access.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(acid) SERVAUTH(`*resource*`)`

Prefix length

One to eight characters

Capacity of list

One to fives prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`) SERVAUTH(`*resource*`)`

Prefix length

One to 64 characters

Capacity of list

One to five prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The SERVAUTH ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The SERVAUTH ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- SERVAUTH(OWN) authority via the TSS ADMIN function to ADD or REMOVE ownership of SERVAUTH resources from ACIDs
- SERVAUTH(XAUTH) authority via the TSS ADMIN function to permit or revoke access to SERVAUTH resources

The administrator can use any of the following methods to control access to SERVAUTH resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: SERVAUTH resource class

This example protects resources by using CA Top Secret, by assigning ownership to the Corporate Department (owning acid):

```
TSS ADDTO(CORPORAT) SERVAUTH(EZB.)
```

The administration may now PERMIT access to users or profiles that requires access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) SERAUTH(EZB.)
```

This example permits users in the Technical Services Department to access TCP/IP stack:

```
TSS PERMIT(acid) SERAUTH(EZB.STACKACCESS.)
                 ACCESS(READ)
```

This example revokes access:

```
TSS REVOKE(acid) SERVAUTH(EZB.STACKACCESS.)
```

SERVER Resource Class—Secure z/OS SERVER Objects

Valid on z/OS.

Use SERVER to secure SERVER for z/OS objects.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) SERVER(`*`server`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) SERVER(`*`server`*`)`

Prefix length

One to sixteen characters

Capacity of list

One to eight prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The SERVER ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The SERVER ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- SERVER (OWN) authority to ADD or REMOVE ownership of SERVER resources from ACIDs
- SERVER(XAUTH) authority to permit or revoke access to SERVER resources

The administrator can use any of the following methods to control access to SERVER resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: SERVER resource class

This example protects the resource by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) SERVER(SERVER1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) SERVER(SERVER1)
```

This example permits users in the Technical Services Department to access SERVER1.XXXXXXXX on Fridays only:

```
TSS PERMIT(acid) SERVER(SERVER1.XXXXXXXX)
                 DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access SERVER1.XXXXXXXX.

```
TSS PERMIT(acid) SERVER(SERVER1.XXXXXXXX)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) SERVER(SERVER1.XXXXXXXX)
```

SMESSAGE Resource Class—Secure TSO Messages

Valid on z/OS.

Use SMESSAGE to secure the transmission of messages from one TSO user to other TSO users via the TSO SEND command.

This resource class has the following format:TSSADDTO/REMOVE

`TSS ADDTO(`*`acid`*`) SMESSAGE(`*`receiving-id`*`,`*`receiving-id`*`...)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) SMESSAGE(`*`receiving-id`*`,`*`receiving-id`*`...)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The SMESSAGE ACID types User Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The SMESSAGE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- SMESSAGE(OWN) authority to ADD or REMOVE control of message routing to a user by ACIDs
- SMESSAGE(XAUTH) authority to permit or revoke the ability to route messages to target userids

The administrator can:

- Specify the following access levels: NONE and READ.
- Use any of the following methods to control access to SMESSAGE resources: Expiration, Facility, Time/Day, and Actions.

Examples: SMESSAGE resource class

This example gives TSOADM1 ownership of a few TSO userids so he can control routing of messages to these userids:

```
TSS ADDTO(TSOADM1) SMESSAGE(TSOJOE1,TSU34A,ANYID)
```

This example removes ownership:

```
TSS REMOVE(TSOADM1) SMESSAGE(TSOJOE1,TSU34A,ANYID)
```

This example permits ACID CHATTY to send messages to userid TESTID:

```
TSS PERMIT(CHATTY) SMESSAGE(TESTID) ACCESS(READ)
```

This example stops routing of messages to userid NOBOTHER:

```
TSS PERMIT(ALL) SMESSAGE(NOBOTHER) ACCESS(NONE)
```

SOMDOBJS Resource Class—Secure z/OS SOMDOBJS Objects

Valid on z/OS.

Use SOMDOBJS to secure SOMDOBJS for z/OS objects

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) SOMDOBJS(`*`object`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) SOMDOBJS(`*`object`*`)`

Prefix length

One to sixteen characters

Capacity of list

One to eight prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The SOMDOBJS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The SOMDOBJS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- SOMDOBJS (OWN) authority to ADD or REMOVE ownership of SOMDOBJS resources from ACIDs
- SOMDOBJS(XAUTH) authority to permit or revoke access to SOMDOBJS resources

The administrator can use any of the following methods to control access to SOMDOBJS resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: SOMDOBJS resource class

This example protects the resource by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) SOMDOBJS(OBJECT1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) SOMDOBJS(OBJECT1)
```

This example permits users in the Technical Services Department to access OBJECT1.XXXXXXXX on Fridays only:

```
TSS PERMIT(TECHPROF) SOMDOBJS(OBJECT1.XXXXXXXX)
                     DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access OBJECT1.XXXXXXXX

```
TSS PERMIT(TECHPROF) SOMDOBJS(OBJECT1.XXXXXXXX)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) SOMDOBJS(OBJECT1.XXXXXXXX)
```

SPI Resource Class—Secure CEMT and CICS

Valid on z/OS.

Use SPI to secure:

- CEMT INQUIRE, SET, and PERFORM
- EXEC CICS INQUIRE and CICS
- EXEC CICS ENABLE, DISABLE, EXTRACT, and COLLECT STATISTICS
- EXEC CICS SPOOLOPEN

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) SPI(`*`command keyword`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands (with the SPI keyword) CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The SPI ACID types User, Profile, Department, Division, and Zone
- SPI(OWN) authority to ADD or REMOVE SPI resources to ACIDs

For information on the syntax and usage of IBM CEMT and EXEC CICS commands, see IBM's *CICS-Supplied Transactions and CICS Customization Guides.*

Equivalents for CEMT INQUIRE and SET

The SPI keyword equivalents for CEMT INQUIRE and SET are:

ATOMSERV

An Atom service, feed, collection, or category document that CICS can deliver to a web client over HTTP.

AUTOINST

Automatic installation of terminals.

AUXTRACE

Indicates whether auxiliary trace is active, which auxiliary data set traces are active and whether on not auxiliary trace data sets are open or closed.

BUNDLE

A unit of deployment for an application.

CONNECTI

Contains the four-character "sysid" used in the TCT (Terminal Control Table) for Intersystem Communication (ISC) or Inter-region Communication (IRC).

CONTROL

The control unit associated with a terminal.

DATASET

The eight-character name used in the FCT (File Control Table).

DLIDATBA

The eight-character name used for DL/I (DDIR control block).

DUMP

Indicates whether the dump data set is opened or closed.

DUMPOPTI

Selects system dump.

CAPTURE/EVENTBIN

Installing the BUNDLE resource generates one or more EVENTBINDING and CAPTURESPEC resources.

EVENTPRO

Global statistics for event processing.

IRBATCH

Identifies the batch job sharing data with CICS for batch regions connected to CICS via IRC (Inter-region communication).

IRC

Indicates whether the IRC facility is active.

JOURNAL

The journal number.

JVMSERVE

The JVMSERVER resource defines the location of the JVM profile and the Language Environment options that are required to create a Language Environment enclave and a JVM server in a CICS region.

LINE

The name of any line terminal.

MODENAME

The eight-character name for a group of sessions.

MQINI

The MQINI resource represents the default initiation queue that is specified by the INITQNAME attribute of the MQCONN resource.

NETNAME

The name defining the remote system to the network.

PITRACE

Indicates whether program isolation trace is active.

PROGRAM

The eight-character program name defined in the PPT (Program Control Table).

QUEUE

Indicates the four-character queue name used in the DCT (Destination Control Table).

SYSTEM

System parameters.

TASK

Indicates the number of the task identifier.

TCLASS

Indicates the class to which the task belongs.

TERMINAL

The four-character terminal used in the TCT (Terminal Control Table).

TRACE

Indicates whether the trace is active.

TRANSACT

The four-character transaction name used in the PCT (Program Control Table).

VOLUME

The six-character volume serial number.

VTAM

Indicates a connection with CICS.

XMLTRANS

If you install the BUNDLE resource, CICS dynamically creates an XMLTRANSFORM resource that defines where the XML binding and schema are located.

SPI Equivalents for CEMT PERFORM

The SPI keyword equivalents for CEMT PERFORM are:

RECONNEC

Indicates that CICS is reconnected after a failure.

RESET

Synchronizes the system date and time with your CICS date and time.

SHUTDOWN

Indicates that CICS is to be shutdown.

SNAP

Gives a picture of your CICS system.

SPI Equivalents for CEMT ADD and REMOVE

The SPI keywords equivalents for CEMT ADD and REMOVE are:

VOLUME

The six-character volume serial number.

SPI Equivalents for EXEC CICS INQUIRE and SET

The SPI keyword equivalents for EXEC CICS INQUIRE and SET are:

ATOMSERV

An Atom service, feed, collection, or category document that CICS can deliver to a web client over HTTP.

BUNDLE

A unit of deployment for an application.

CONNECTI

Contains the four-character "sysid" used in the TCT (Terminal Control Table), for Intersystem Communication (ISC) or Inter-region Communication (IRC).

DATASET

The eight-character name used in the FCT (File Control Table).

CAPTURE/EVENTBIN

Installing the BUNDLE resource generates one or more EVENTBINDING and CAPTURESPEC resources.

EVENTPRO

Global statistics for event processing.

FILE

The eight-character data set name in the FCT.

JVMSERVE

The JVMSERVER resource defines the location of the JVM profile and the Language Environment options that are required to create a Language Environment enclave and a JVM server in a CICS region.

MODENAME

The eight-character name for a mode group defined for a specific system connection.

MQINI

The MQINI resource represents the default initiation queue that is specified by the INITQNAME attribute of the MQCONN resource.

PROGRAM

The eight-character program name defined in the PPT (Program Control Table).

SYSTEM

Indicates the current values of your system parameters.

TERMINAL

The four-character terminal name used in the TCT (Terminal Control Table).

TRANSACT

The four-character transaction name used in the PCT (Program Control Table).

XMLTRANS

If you install the BUNDLE resource, CICS dynamically creates an XMLTRANSFORM resource that defines where the XML binding and schema are located.

SPI Equivalents for EXEC CICS ENABLE, DISABLE, EXTRACT, COLLECT STATISTICS

The SPI keyword equivalents for EXEC CICS ENABLE, DISABLE, EXTRACT, and COLLECT STATISTICS are:

ENABLE

Specifies that all or part of the enable sequence for an exit program is performed.

DISABLE

Specifies that all or part of the disable sequence for an exit program is performed.

EXTRACT

Indicates that data is extracted from CICS control blocks.

COLLECT STATISTICS

Collects current statistics for an individual resource or collects global statistics on a CICS defined class of resources.

SPI Equivalents for EXEC CICS SPOOLOPEN

The SPI keyword equivalents for EXEC CICS SPOOLOPEN is:

JESSPOOL

Opens a spool report for input to CICS and reads existing spool data sets using external writer names for the userid.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) SPI(command keyword)
                 ACCESS(acction keyword(s))
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The SPI ACID types User, Profile, Department, Division, and Zone
- SPI(XAUTH) authority to permit or revoke access to SPI resources

For information on the syntax and usage of IBM CEMT and EXEC CICS commands, see IBM's *CICS-Supplied Transactions and CICS Customization* guides and the *Implementation: CICS Guide.*

SPI Access Levels for CEMT INQUIRE and SET

The following table details specific SPI access levels:

CEMT Action	SPI Acess Level
ADD	SET
INQUIRE	INQUIRE
PERFORM	PERFORM
REMOVE	REMOVE
SET	SET

SPI Access Levels for EXEC CICS INQUIRE and SET

The following table details specific SPI access levels:

EXEC CICS Command	SPI Access Level
INQUIRE	INQUIRE
SET	SET

SPI Access Levels for EXEC CICS ENABLE, DISABLE, EXTRACT, COLLECT STATISTICS

The following table details specific SPI access levels:

Command Function	SPI Access Level
ENABLE	SET
DISABLE	SET
EXTRACT	READ
COLLECT STATISTICS	COLLECT

SPI Access Levels for EXEC CICS SPOOLOPEN

The following table details specific SPI access levels:

Command Options	SPI Access Level
INPUT	SET
OUTPUT	SET

Examples: SPI resource class

This example adds the SPI SYSTEM resource to a user:

```
TSS ADDTO(USER01) SPI(SYSTEM)
```

This example adds the equivalent SPI CONNECTION keyword to a user (using the CICS CEMT command):

```
TSS ADDTO(USER01) SPI(CONNECTION)
```

This example adds the equivalent SPI ENABLE keyword to a user (using the EXEC CICS command):

```
TSS ADDTO(USER01) SPI(ENABLE)
```

This example adds the equivalent SPI SPOOLOPEN keyword to a user (using the EXEC CICS command):

```
TSS ADDTO(USER01) SPI(JESSPOOL)
```

This example permits access to the SPI SYSTEM resource:

```
TSS PERMIT(USER01) SPI(SYSTEM) ACCESS(INQUIRE)
```

This example permits access to the SPI CONNECTION resource, (using the CICS CEMT command):

```
TSS PERMIT(USER01) SPI(CONNECTION) ACCESS(SET)
```

This example permits access to the SPI ENABLE resource (using the EXEC CICS command):

```
TSS PERMIT(USER01) SPI(ENABLE) ACCESS(SET)
```

This example permits access to the SPI SPOOLOPEN resource (for INPUT or OUTPUT):

```
TSS PERMIT(USER01) SPI(JESSPOOL) ACCESS(SET)
```

STATION Resource Class—Secure CADISPATCH and CASCHEDULER

Valid on z/OS.

Use STATION to secure STATION for CA-DISPATCH and CA-SCHEDULER.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*acid*`) STATION(`*resource*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`) STATION(`*resource*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The STATION ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The STATION ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- STATION (OWN) authority to ADD or REMOVE ownership of STATION resources from ACIDs
- STATION(XAUTH) authority to permit or revoke access to STATION resources

The administrator can use any of the following methods to control access to STATION resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: STATION resource class

This example protects the resource by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) STATION(ABCDEFGH)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) STATION(ABCDEFGH)
```

This example permits users in the Technical Services Department to access ABCDEFGH on Fridays only:

```
TSS PERMIT(TECHPROF) STATION(ABCDEFGH)
                     DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access ABCDEFGH:

```
TSS PERMIT(TECHPROF) STATION(ABCDEFGH)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) STATION(ABCDEFGH)
```

STORCLAS Resource Class—Secure SMS Storage Classes

Valid on z/OS.

Use STORCLAS to secure SMS storage classes as defined by the storage administrator.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*acid*`) STORCLAS(`*storage classes*`,...)`

Prefix length

One to eight characters

Capacity of list

Five storage classes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`) STORCLAS(`*storage classes*`,...)`

Prefix length

One to eight characters

Capacity of list

Five storage classes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The STORCLAS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The STORCLAS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- STORCLAS(OWN) authority to protect user access to specific storage classes
- STORCLAS(XAUTH) authority to permit or revoke access to storage classes

The administrator can use any of the following methods to control access to STORCLAS: Expiration, Facility, Time/Day, and Actions.

Examples: STORCLAS resource class

This example protects storage class of PRODSTOR:

```
TSS ADDTO(DEPT02) STORCLAS(PRODSTOR)
```

This example removes ownership:

```
TSS REMOVE(DEPT02) STORCLAS(PRODSTOR)
```

This example allows an ACID to use the PRODSTOR storage class:

```
TSS PERMIT(USER02) STORCLAS(PRODSTOR)
```

This example revokes access:

```
TSS REVOKE(USER02) STORCLAS(PRODSTOR)
```

SUBSCHEM Resource Class—Secure CAIDMS Subschema Names

Valid on z/OS.

Use SUBSCHEM to secure CA-IDMS subschema names.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) SUBSCHEM(`*`subschum`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) SUBSCHEM(`*`subschum`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The SUBSCHEM ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The SUBSCHEM ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- SUBSCHEM(OWN) authority to ADD or REMOVE ownership of SUBSCHEMs from ACIDs
- SUBSCHEM(XAUTH) authority to permit or revoke access to SUBSCHEMs

The administrator can use any of the following methods to control access to SUBSCHEMs: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: SUBSCHEM Resource Class—Secure

This example assigns ownership of the subschema, PERDB, to the Personnel Department:

```
TSS ADDTO(PERSDP) SUBSCHEM(PERDB)
```

This example removes ownership:

```
TSS REMOVE(PERSDP) SUBSCHEM(PERDB)
```

This example allows USER01 access to the division's subschema:

```
TSS PERMIT(USER01) SUBSCHEM(PAYFILE)
```

This example revokes access:

```
TSS REVOKE(USER01) SUBSCHEM(PAYFILE)
```

SURROGAT Resource Class—Restrict Preset Security

Valid on z/OS and z/VM.

Use SURROGAT to restrict installation of pre-set security for terminals to specific ACIDs. SURROGAT can only be used with terminals running under CICS 4.1.

The SURROGAT keyword is used to establish access authorizations and restrictions.

This resource class has the following format for ADDTO/REMOVE:

```
TSS ADDTO(acid) SURROGAT(acidname.DFHINSTAL)
```

Prefix length

Two to 17 characters

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) SURROGAT(acidname.DFHINSTAL)
```

Prefix length

Two to 44 characters

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The SURROGAT ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The SURROGAT ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- SURROGAT(OWN) authority to ADD or REMOVE terminals from ACIDs
- SURROGAT(XAUTH) authority to permit or revoke access to terminals that are owned

The administrator can use any of the following methods to control access to surrogate terminals: Expiration, Facility, Time/Day, and Actions.

Example: SURROGAT resource class

This example installs a terminal with a pre-set userid of WAREHOUS, an ACID named WAREHOUS must be defined to CA Top Secret with permission to the CICS Facility and any appropriate resources:

```
TSS ADDTO(CICSDEPT) SURROGAT(WAREHOUS.DFHINSTAL)
```

This example permits the user defining the terminal access to the SURROGAT resource for WAREHOUS:

```
TSS PERMIT(CICSADM) SURROGAT(WAREHOUS.DFHINSTL)
                    ACCESS(READ)
```

SYSAUTO Resource Class—IBM Automation Control for z/OS Resources

Valid on z/OS.

Use SYSAUTO to control access to z/OS resources for IBM Automation Control.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) SYSAUTO(ZOSRES)
```

Prefix length

2 - 26 characters.

Capacity of list

1 - 5 prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) SYSAUTO(ZOSRES)
```

Resource length

1 - 246 characters.

This resource class is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS.
- The ACID types User, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE.
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE.

- SYSAUTO(OWN) authority to ADD or REMOVE ownership of SYSAUTO resources from ACIDs.
- SYSAUTO(XAUTH) authority to PERMIT or REVOKE access to SYSAUTO resources.

The administrator can use any of the following methods to control access to SYSAUTO resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: SYSAUTO resource class

- This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department ACID:

  ```
  TSS ADDTO(AUTODEPT) SYSAUTO(ZOSRES)
  ```

  The administrator can now PERMIT access to users or profiles that require access.
- This example removes ownership:

  ```
  TSS REMOVE(AUTODEPT) SYSAUTO(ZOSRES)
  ```
- This example permits users to access SYSAUTO(ZOSRES):

  ```
  TSS PERMIT(TECHUSER) SYSAUTO(ZOSRES)
  ```

SYSCONS Resource Class—Secure IBM Consoles

Valid on z/OS.

Use SYSCONS to secure IBM consoles. Equivalent to the IBM CONSOLE resource class.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) SYSCONS(`*`id`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) SYSCONS(`*`id`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, WHOHAS, and WHOOWNS
- The SYSCONS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The SYSCONS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- SYSCONS(OWN) authority to ADD or REMOVE ownership of SYSCONS from ACIDs
- SYSCONS(OWN) authority to permit or revoke access to SYSCONS

The administrator can use any of the following methods to control access to SYSCONS: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: SYSCONS Resource Class

This example adds consoles 01 through 04 to the PERSDP department:

```
TSS ADDTO(PERSDP) SYSCONS(01,02,03,04)
```

This example removes ownership:

```
TSS REMOVE(PERSDP) SYSCONS(01,02,03,04)
```

This example allows USER01 access to the 01 console:

```
TSS PERMIT(USER01) SYSCONS(01)
```

This example revokes access:

```
TSS REVOKE(USER01) SYSCONS(01)
```

SYSMVIEW Resource Class—Secure SystemView Segments

Valid on z/OS.

Use SYSMVIEW to secure a resource class named SYSMVIEW that protects SystemView segments.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) SYSMVIEW(SystemView-segment-name)
```

Prefix length

One to eight characters

Capacity of list

One to five SystemView-segment-names per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) SYSMVIEW(SystemView-segment-name)
                 APPLDATA(applata)
                 SCRIPTNAME(scriptname)
                 SCRIPTPARM(scriptparm)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, and REVOKE
- The SYSMVIEW ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The SYSMVIEW ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- SYSMVIEW(OWN) authority to ADD or REMOVE ownership of a SYSMVIEW resource from ACIDs
- SYSMVIEW(XAUTH) authority to permit or revoke access to SYSMVIEWs

Three keywords are allowed with the SystemView segment when permitting it to an ACID:

- APPLDATA
- SCRIPTNAME
- SCRIPTPARM

Examples: SYSMVIEW resource class

This example protects a SystemView segment, A01ITSO.SYSTEMA, by assigning ownership to the Corporate Level Department (CORPDEPT):

```
TSS ADDTO(CORPDEPT) SYSMVIEW(A01ITSO.SYSTEMA)
```

This example removes ownership:

```
TSS REMOVE(CORPDEPT) SYSMVIEW(A01ITSO.SYSTEMA)
```

This example permits a user full access to SYSMVIEW, A01ITSO.SYSTEMA, including APPLDATA, SCRIPTNAME and SCRIPTPARM parameters:

```
TSS PERMIT(USER05) SYSMVIEW(A01ITSO.SYSTEMA)
                       APPLDATA(ptktl)
                       SCRIPTName(tsoscript)
                       SCRIPTParm(tsoparm)
```

This example revokes access:

```
TSS REVOKE(USER05) SYSMVIEW(A01ITSO.SYSTEMA)
```

TCICSTRN Resource Class—Secure CICS TCICSTRN RESCLASS

Valid on z/OS.

Use TCICSTRN to secure TCICSTRN RESCLASS for CICS

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`)TCICSTRN(`*`tranid`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`)TCICSTRN(`*`tranid`*`)`

Prefix length

One to eight characters

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The TCICSTRN ACID types User, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The TCICSTRN ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- TCICSTRN (OWN) authority to ADD or REMOVE ownership of TCICSTRN resources from ACIDs
- TCICSTRN(XAUTH) authority to permit or revoke access to TCICSTRN resources

The administrator can use any of the following methods to control access to TCICSTRN resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: TCICSTRN resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department (owning acid):

```
TSS ADDTO(CORPORAT)TCICSTRN(CEMT)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT)TCICSTRN(CEMT)
```

```
TSS PERMIT(TECHPROF)TCICSTRN(CEMT)
```

This example revokes access:

```
TSS REVOKE(TECHUSER)TCICSTRN(CEMT)
```

TERMINAL Resource Class—Secure Terminals

Valid on z/OS, z/VSE, and z/VM.

Use TERMINAL to secure terminal IDs and PCs used to access the mainframe.

The TERMINAL keyword is used to establish access authorizations and restrictions.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`)TERMINAL(`*`terminal`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command

Terminal restriction is used to restrict Automatic Terminal Signon ACIDs from being used at another terminal.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The TERMINAL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The TERMINAL ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- TERMINAL(OWN) authority to ADD or REMOVE terminals from ACIDs
- TERMINAL(XAUTH) authority to permit or revoke access to TERMINALs

The administrator can use any of the following methods to control access to TERMINALs: Expiration, Facility, Time/Day, and Actions.

Note: All terminals is protected by setting DEFPROT on the resource class TERMINAL.

Defining Terminals

This section details the options for defining terminals

Terminal Definitions for z/VM:

The following table provide instructions and examples on how to specify prefixes for z/VM:

Type	Prefix	Example
Locally attached	GRAF plus four-character local address	TSS ADDTO(BUDDEPT) TERMINAL(GRAF02BA)
Remotely attached VM-controlled network terminals	NETW plus four-character resource ID	TSS ADDTO(CORP) TERMINAL(NETW0301)
Logical devices	LDEV plus four-character address of logical device which is arbitrarily defined.	TSS ADDTO(CORPNET) TERMINAL(LDEV1234)
VTAM/SNA	Eight-character LU name	TSS ADDTO(FINDEPT) TERMINAL(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.

Terminal Definitions for z/OS

The following table provide instructions and examples on how to specify prefixes for z/OS:

Type	Prefix	Example
JES Readers:	This keyword uses names known to JES, as the following entries show.	n/a
RJE	REMOTE #@ READER# Rnn.RDnn	TSS ADDTO(BUDDEPT) TERMINAL(R12.RD1) Assigns remote 12, reader 1 to the Budget Department
NJE	Symbolic Name Node # @ Remote # Nnn.Rnn	TSS ADDTO(CORPNET) TERMINAL(PHILA) TSS ADDTO(CORPNET) TERMINAL(N2.R4)

Type	Prefix	Example
Local	READER1	TSS ADDTO(CORPNET) TERMINAL(READER1)
Terminals	This keyword uses the name known to TCAM or VTAM via TP monitor definitions. *ALL*	To protect VTAM terminals (cluster name TSONxxx), enter: TSS ADDTO(CORP) TERMINAL(TSON) To allow a user to access all protected terminals, perform the following: 1. Assign ownership of *ALL* to the MSCA: TSS ADDTO(MSCA) TERMINAL(*ALL*) 2. PERMIT the user to access all terminals: TSS PERMIT(user) TERMINAL(*ALL*)

Terminal Definitions for PCs

The following table provide instructions and examples on how to specify prefixes for PCs:

Type	Prefix	Example
PC	Eight-character LU name	TSS ADDTO(DEVDEPT) TERMINAL(xxxxxxxx)

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid)TERMINAL(prefix(es))
```

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command

Examples: TERMINAL resource class

This example gives the Finance Department, (FINDEPT), ownership of a local terminal in the personnel office (address K61L1234):

```
TSS ADDTO(FINDEPT)TERMINAL(K61L1234)
```

This example removes ownership:

```
TSS REMOVE(FINDEPT)TERMINAL(K61L1234)
```

This example permits all users connected to the PAYROLL profile (PAYPROF1) access to a local terminal in the personnel office (address K61L1234), from 7:00 to 11:00 am:

```
TSS PERMIT(PAYPROF1)TERMINAL(K61L1234)TIMES(07,11)
```

This example revokes access:

```
TSS REVOKE(PAYPROF1)TERMINAL(K61L1234)
```

TIMS Resource Class—Secure IMS TIMS

Valid on z/OS.

Use TIMS to secure TIMS for IMS resource.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`)TIMS(`*`resource`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`)TIMS(`*`resource`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The TIMS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The TIMS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- TIMS (OWN) authority to ADD or REMOVE ownership of TIMS resources from ACIDs
- TIMS(XAUTH) authority to permit or revoke access to TIMS resources

The administrator can use any of the following methods to control access to TIMS resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: TIMS resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT)TIMS(IMS1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT)TIMS(IMS1)
```

This example permits users in the Technical Services Department to access IMS1 on Fridays only:

```
TSS PERMIT(TECHPROF)TIMS(IMS1) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access IMS1:

```
TSS PERMIT(TECHPROF)TIMS(IMS1)
```

This example revokes access:

```
TSS REVOKE(TECHPROF)TIMS(IMS1)
```

TOTAL Resource Class—Secure TOTAL for CINCOM

Valid on z/OS.

Use TOTAL to secure TOTAL for CINCOM product called TOTAL.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`)TOTAL(`*`resource`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`)TOTAL(`*`resource`*`)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The TOTAL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The TOTAL ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- TOTAL (OWN) authority to ADD or REMOVE ownership of TOTAL resources from ACIDs
- TOTAL(XAUTH) authority to permit or revoke access to TOTAL resources

The administrator can use any of the following methods to control access to TOTAL resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: TOTAL resource class

This example protects the resource by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT)TOTAL(CINCOM)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT)TOTAL(CINCOM)
```

This example permits users in the Technical Services Department to access CINCOM on Fridays only:

```
TSS PERMIT(TECHPROF)TOTAL(CINCOM) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access CINCOM:

```
TSS PERMIT(TECHPROF)TOTAL(CINCOM)
```

This example revokes access:

```
TSS REVOKE(TECHPROF)TOTAL(CINCOM)
```

TSOACCT Resource Class—Secure TSO Logon Account Codes

Valid on z/OS.

Use TSOACCT to secure account codes that are used during TSO logon.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`)TSOACCT(`*`code`*`)`

Prefix length

One to eight characters

Capacity of list

five accounts per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`)TSOACCT(`*`prefix(es)`*`)`

Prefix length

One to forty-four characters

Capacity of list

five accounts per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The TSOACCT ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The TSOACCT ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- TSOACCT(OWN) authority to protect user access to specific accounting codes
- TSOACCT(XAUTH) authority to permit or revoke access to accounting codes

The administrator can use any of the following methods to control access to TSO accounting fields: Expiration, Facility, Time/Day, and Actions.

Examples: TSOACCT resource class

This example protects TSOACCT of A044221:

```
TSS ADDTO(ACCTDEPT)TSOACCT(A044221)
```

This example removes ownership:

```
TSS REMOVE(ACCTDEPT)TSOACCT(A044221)
```

This example allows an ACID to use A044221:

```
TSS PERMIT(USERA)TSOACCT(A044221)
```

This example revokes access:

```
TSS REVOKE(USERA)TSOACCT(A044221)
```

TSOAUTH Resource Class—Secure TSO User Attributes

Valid on z/OS.

Use TSOAUTH to secure TSO user attributes.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`)TSOAUTH(`*`authority...`*`)`

Prefix length

One to eight characters

Capacity of list

Five authorities per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`)TSOAUTH(`*`prefix(es)`*`)`

Prefix length

One to eight characters

Capacity of list

Five authorities per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The TSOAUTH ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- TSOAUTH ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- TSOAUTH(OWN) authority to protect user access to specific attributes.
- TSOAUTH(XAUTH) authority to permit or revoke access to accounting codes

The administrator can use any of the following methods to control access to TSO authorities: Expiration, Facility, Time/Day, and Actions.

Examples: TSOAUTH resource class

This example protects the user of TSO MOUNT authority:

```
TSS ADDTO(SYSDEPT)TSOAUTH(MOUNT)
```

This example removes ownership:

```
TSS REMOVE(SYSDEPT)TSOAUTH(MOUNT)
```

This example allows an ACID to use the OPER command:

```
TSS PERMIT(USERA)TSOAUTH(OPER)
```

This example revokes access:

```
TSS REVOKE(USERA)TSOAUTH(OPER)
```

TSOPRFG Resource Class—Secure TSO Performance Groups

Valid on z/OS.

Use TSOPRFG to secure TSO performance groups.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`)TSOPRFG(`*`performance group`*`,...)`

Prefix length

One to three digits

Capacity of list

five performance groups per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`)TSOPRFG(`*`performance group`*`,...)`

Prefix length

One to three digits

Capacity of list

Five performance groups per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The TSOPRFG ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The TSOPRFG ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- TSOPRFG(OWN) authority to protect user access to specific performance groups.
- TSOPRFG(XAUTH) authority to permit or revoke access to a TSO performance group

The administrator can use any of the following methods to control access to TSO performance groups: Expiration, Facility, Time/Day, and Actions.

Examples: TSOPRFG resource class

This example protect TSOPRFG of 001:

```
TSS ADDTO(USER01)TSOPRFG(001)
```

This example removes ownership:

```
TSS REMOVE(USER01)TSOPRFG(001)
```

This example allows an ACID to use the performance group of PRFG001:

```
TSS PERMIT(USER01)TSOPRFG(001)
```

This example revokes access:

```
TSS REVOKE(USER01)TSOPRFG(001)
```

TSOPROC Resource Class—Secure TSOP Logon Procs

Valid on z/OS.

Use TSOPROC to secure PROCs for TSO logon.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*acid*`)TSOPROC(`*proc*`,...)`

Prefix length

One to eight characters

Capacity of list

Five TSOPROCs per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`)TSOPROC(`*proc*`,...)`

Prefix length

One to eight characters

Capacity of list

Five TSOPROCs per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The TSOPROC ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The TSOPRFG ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- TSOPROC(OWN) authority to protect user access to specific TSO logon procs
- TSOPROC(XAUTH) authority to permit or revoke access to a TSO logon procs

The administrator can use any of the following methods to control access to a TSO logon proc: Expiration, Facility, Time/Day, and Actions.

Examples: TSOPROC resource class

This example protects logon proc of SMPEPROC:

```
TSS ADDTO(SYSDEPT)TSOPROC(SMPEPROC)
```

This example removes ownership:

```
TSS REMOVE(SYSDEPT)TSOPROC(SMPEPROC)
```

This example allows an ACID to use SMPEPROC:

```
TSS PERMIT(SYSUSER)TSOPROC(SMPEPROC)
```

This example revokes access:

```
TSS REVOKE(SYSUSER)TSOPROC(SMPEPROC)
```

TST Resource Class—Secure CICs Temporary Storage Table Names

Valid on z/OS.

Use TST to secure CICS Temporary Storage Table names.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`)TST(`*`name`*`)`

Prefix length

One to sixteen characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`)TST(`*`prefix(es)`*`) ACCESS(`*`access levels`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The TST ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The TST ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- TST(OWN) authority to ADD or REMOVE TSTs from ACIDs
- TST(XAUTH) authority to permit or revoke access to TSTs

The administrator can:

- Specify the access levels: INQUIRE, SET, ALL, NONE, PURGE, READ, REPLACE, and WRITE. If access in not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to TSTs: Expiration, Facility, Pathing, Time/Day, and Actions.

Example: TST resource class

This example assigns ownership of a TST to Department 5:

```
TSS ADDTO(DEPT5)TST(XCOM7)
```

This example removes ownership:

```
TSS REMOVE(DEPT5)TST(XCOM7)
```

This example permits a CICS user to access restricted temporary storage when using the CICSTEST facility:

```
TSS PERMIT(SYST3)TST(XCOM7) ACCESS(ALL)
                            FACILITY(CICSTEST)
```

This example revokes access:

```
TSS REVOKE(SYST3)TST(XCOM7)
```

UR1/UR2 Resource Class—Secure Resource Restrictions

Valid on z/OS, z/VSE, and z/VM.

Use UR1/UR2 to secure general installation-defined resource restrictions as account codes.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) UR1(oper,oper,...)|UR2(oper,oper,...)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) UR1(p-fix)|UR2(p-fix)
                 ACCESS(access levels)
```

Prefix length

One to forty-four characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The UR1/UR2 keywords are used with the ACID types: User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- UR1/UR2(OWN) authority to ADD or REMOVE UR1/UR2 resources from ACIDs
- UR1/UR2(XAUTH) authority to permit or revoke access to UR1/UR2 resources

The administrator can:

- Specify the access levels: ALL, NONE, READ, WRITE, and UPDATE. If access is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to user-defined resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Customization

This keyword uses:of the UR1 or UR2 keyword requires prior customization of application programs or system exits to invoke the VM and z/OS security interface such that, it will call CA Top Secret to verify access to installation-defined resources.

Resource Class—Secure

An installation-defined resource type, such as account codes, is identified as being a class 1 (UR1) or a class 2 (UR2) user owned resource.

Ownership

Once the resource classes are defined, the administrator can use the UR1 keyword to assign ownership of up to five UR1 resource prefixes.

Customization

This keyword uses:of the UR1 or UR2 keyword requires prior customization using the z/OS and VM security interface such that it calls CA Top Secret to verify access to installation-defined resource.

Resource Class—Secures

An installation-defined resource type, such as account codes, is identified as being a class 1 (UR1) or a class 2 (UR2) user owned resource.

Examples: UR1/UR2 resource class

This example assigns ownership of an account number to Corporate:

```
TSS ADDTO(CORPORAT) UR1(ACCT788I)
```

The administrator may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) UR1(ACCT788I)
```

This example permits a group of users all access to a set of account numbers:

```
TSS PERMIT(CLKPROF) UR2(ACCT7889,ACCT4567) ACCESS(ALL)
```

This example revokes access:

```
TSS REVOKE(CLKPROF) UR2(ACCT7889,ACCT4567)
```

USRCLASS Resource Class—Secure Resource Classes

Valid on z/OS and z/VM.

Use USRCLASS to secure resource classes used by other CA Product interfaces.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) USRCLASS(`*`prefix(es)`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) USRCLASS(`*`prefix(es)`*`)`

Prefix length

One to forty-four characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The USRCLASS ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The USRCLASS ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- USRCLASS(OWN) authority to ADD or REMOVE authority to control access of CA product specific resources owned
- USRCLASS(XAUTH) authority to permit or revoke the authority to CA product specific commands, functions and programs

The administrator can use any of the following methods to control access to USRCLASS resources: Expiration, Facility, Time/Day, and Actions.

Examples: USRCLASS resource class

This example gives SYSPRG ownership of the USRCLASS named BACKUP:

```
TSS ADDTO(SYSPRG) USRCLASS(BACKUP)
```

This example removes ownership:

```
TSS REMOVE(SYSPRG) USRCLASS(BACKUP)
```

This example permits ACID, USER01, the ability to use the USRCLASS BACKUP:

```
TSS PERMIT(VMUSER1) USRCLASS(BACKUP)
```

To enable all users to issue the USRCLASS RESTORE:

```
TSS PERMIT(ALL) USRCLASS(RESTORE)
```

VMANAPPL Resource Class—Secure CAVMAN VMANAPPL

Valid on z/OS.

Use VMANAPPL to secure VMANAPPL for CAVMAN.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) VMANAPPL(APPLABCD)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command

`TSS PERMIT(`*`acid`*`) VMANAPPL(APPLABCD)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The VMANAPPL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The VMANAPPL ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- VMANAPPL (OWN) authority to ADD or REMOVE ownership of VMANAPPL resources from ACIDs
- VMANAPPL(XAUTH) authority to permit or revoke access to VMANAPPL resources

The administrator can use any of the following methods to control access to VMANAPPL resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Example: VMANAPPL resource class

This example protects the resource by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) VMANAPPL(APPLABCD)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) VMANAPPL(APPLABCD)
```

This example permits users in the Technical Services Department to access APPLABCD on Fridays only:

```
TSS PERMIT(TECHPROF) VMANAPPL(APPLABCD) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access APPLABCD.

```
TSS PERMIT(TECHPROF) VMANAPPL(APPLABCD)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) VMANAPPL(APPLABCD)
```

VMCF Resource Class—Secure VMCF Communication Targets

Valid on z/VM.

Use VMCF to secure VMCF communication targets.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) VMCF(userid,userid,userid,...)
```

Prefix length

One to eight characters

Capacity of list

One to five userids per TSS command

Generic prefixing allows the administrator to group a set of similar resource names together, and define them by generic prefix.

Note: The communication targets control the ability of users to issue a VMCF send to the virtual machine designated by the resource name.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) VMCF(oper,oper,oper,...)
```

Full name

One to 13 characters

Capacity of list

One to five userids per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The VMCF ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The VMCF ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- VMCF(OWN) authority to ADD or REMOVE ownership from ACIDs
- VMCF(XAUTH) authority to permit or revoke VMCF access to virtual machines

The administrator can use any of the following methods for restricting access control: Expiration, Time/Day, and Actions.

Examples: VMCF resource class

To protect operations personnel administering CA-Scheduler via the VMCF interface:

```
TSS ADDTO(OPERPROF) VMCF(SCHEDCMS)
```

This example removes ownership:

```
TSS REMOVE(VTERMDPT) VMCF(VTERM7)
```

This example permits a group of systems programmers to send commands to the SMART virtual machine via VMCF:

```
TSS PERMIT(SYSPROF) VMCF(SMART)
```

This example revokes access:

```
TSS REVOKE(SYSPROF) VMCF(SMART)
```

VMDEV Resource Class—Secure z/VM Real Devices

Valid on z/OS.

Use VMDEV to secure access to z/VM real devices.

When used with TSS ADDTO or REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) VMDEV(`*`device`*`)`

Prefix length

2 to 26 characters.

Capacity of list

1-8 prefixes per TSS command.

When used with TSS PERMITor REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) VMDEV(`*`device`*`)`

Prefix length

1 to 246 characters

Capacity of list

1-8 prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The VMDEV ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADDTO or REMOVE
- The VMDEV ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT or REVOKE
- VMEDV(OWN) authority to ADDTO or REMOVE ownership of VMDEV resources from ACIDs
- VMDEV(XAUTH) authority to permit or revoke access to VMDEV resources

An administrator can use any of the following methods to control access to VMDEV resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: VMDEV resource class

This example protects the resource by assigning ownership to the Corporate Department. After ownership is assigned, the administrator can PERMIT access to users or profiles.

```
TSS ADDTO(CORPORAT) VMDEV(DASD)
```

This example permits users in the Technical Services Department to access DISK1234.

```
TSS PERMIT(TECHPROF) VMDEV(DASD.DISK1234)
```

This example revokes access to DISK1234:

```
TSS REVOKE(TECHPROF) VMDEV(DISK1234)
```

VMDIAL Resource Class—Secure Dial Access Virtual Machines

Valid on z/VM.

Use VMDIAL to secure virtual machines with DIAL access security.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) VMDIAL(`*`userid`*`,`*`userid`*`,`*`userid`*`,...)`

Prefix length

One to eight characters

Full name

One to 13 characters

Generic prefixing allows the administrator to group a set of similar resource names together, and define them by generic prefix.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) VMDIAL(`*`oper`*`,`*`oper`*`,...)`

Prefix length

One to 13 characters

Full name

One to 13 characters

Capacity of list

One to five *userids* per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The VMDIAL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The VMDIAL ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- VMDIAL(OWN) authority to ADD or REMOVE ownership from ACIDs

The administrator can:

- Use any of the following methods for restricting dial access to virtual machines: Expiration, Time/Day, and Actions.
- VMDIAL(XAUTH) authority to permit or revoke access to virtual machines

Examples: VMDIAL Resource Class—Secure

This example protects DIAL access to the virtual machine OPER17, by assigning ownership to the Operations Department:

```
TSS ADDTO(OPERDEPT) VMDIAL(OPER17)
```

This example removes ownership:

```
TSS REMOVE(OPERDEPT) VMDIAL(OPER17)
```

This example permits a group in the Payroll Department to dial the virtual machine PAY17:

```
TSS PERMIT(PAYUSER) VMDIAL(PAY17)
```

When PAYUSER dials the virtual machine PAY17, he is prompted for his password. VMDIAL can also restrict which lines an ACID may use when dialing into a virtual machine.

```
TSS PERMIT(PAYUSR2) VMDIAL(PAY17.0041)
```

This example allows PAYUSR2 to dial the PAY17 machine at virtual line 0041 only. This example revokes PAYUSER's dial access permission:

```
TSS REVOKE(PAYUSER) VMDIAL(PAY17.0041)
```

VMMACH Resource Class—Authorize AUTOLOG Users

Valid on z/VM.

Use VMMACH to authorize users to AUTOLOG a virtual machine or logon to it using an alternate ACID.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) VMMACH(`*`machine name`*`)`

Prefix length

One to eight characters

Capacity of list

One to five machine names per TSS command

Generic prefixing allows the administrator to group a set of similar resource names together, and define them by generic prefix.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) VMMACH(`*`machine name`*`)`

Prefix length

One to eight characters

Capacity of list

One to five machine names per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The VMMACH ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The VMMACH ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- VMMACH(OWN) authority to ADD or REMOVE ownership from ACIDs
- VMMACH(XAUTH) authority to permit or revoke access to VMMACHs

The administrator can:

- Specify the access levels: ALL, DEFAULT, AUTOLOG, LOGON, GRPLOGON, SUROGATE, and NONE. If access in not specified, CA Top Secret defaults to NONE access.
- Use any of the following methods for restricting dial access to virtual machines: Autolog, Expiration, Time/Day, and Actions.

Examples: VMMACH resource class

This example restricts the use of the virtual machine BATCH1, by assigning ownership of the Systems Department:

```
TSS ADDTO(SYSDEPT) VMMACH(BATCH1)
```

This example removes ownership:

```
TSS REMOVE(SYSDEPT) VMMACH(BATCH1)
```

This example permits a group of users in the Development Department to autolog the batch controller:

```
TSS PERMIT(PRODEV) VMMACH(BATCH1)
                   ACCESS(AUTOLOG)
```

This example revokes access to BATCH1:

```
TSS REVOKE(PROFDEV) VMMACH(BATCH1)
```

VMMDISK Resource Class—Secure Minidisks

Valid on z/VM.

Use VMMDISK to secure all z/VM minidisks.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) VMMDISK(minidisk)
```

Prefix length

Two to 13 characters.

Full name

Two to 13 characters.

Capacity of list

One to eight minidisks per TSS command.

Generic prefixing allows the administrator to group a set of similar resource names together, and define them by generic prefix.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) VMMDISK(minidisk)
                 ACCESS(access levels)
```

Prefix length

Two to 13 characters.

Full name

Two to 13 characters.

Capacity of list

One to eight minidisks per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The VMMDISK ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The VMMDISK ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- VMMDISK(OWN) authority to ADD or REMOVE ownership of minidisks from ACIDs

The following table indicates the access levels that the administrator can specify:

Level	Link Access
READ	R, RR
WRITE	W
MULTI	M
MREAD	R, RR, M, MR
MWRITE	W, M, MW
SREAD	SR
SWRITE	SW
SMULTI	SM
EREAD	ER
EWRITE	EW
UPDATE	R, RR, W, WR
ALL	Any link is valid
NONE	No link is valid

Note the following:

- Specifying access levels READ and MULTI together implies MREAD access.
- Specifying access levels WRITE and MULTI together implies MWRITE access.

The administrator can use any of the following methods to control access to minidisks: Expiration, Facility, Time/Day, and Actions.

- **Note:** All cuu addresses are assumed to be four digits. For example 190 disk is permitted as MAINT.0190
- VMMDISK(XAUTH) authority to permit or revoke access to minidisks

ALL Access

Ownership implies that the user, profile, or control ACID has an access level of ALL, which allows the ACID to READ and WRITE to the specified minidisk(s).

Ownership

It may not be desirable to grants unlimited access to individual users or profiles. It is recommended that the administrator use the ADDTO command function to assign ownership of minidisks to departments or divisions. The administrator may then authorize full or restricted access to minidisks via the PERMIT command function.

Masking

The VMMDISK resource class supports all masking characters.

Examples: VMMDISK Resource Class—Secure

This example adds a specific VMMDISK, the administrator gives the Inventory Department (INVDEPT) ownership of a 192 minidisk belonging to userid INVEN:

```
TSS ADDTO(INVDEPT) VMMDISK(INVEN.0192)
```

This examp[le removes ownership of minidisks:

```
TSS REMOVE(INVDEPT) VMMDISK(INVEN.0192)
```

This example permits a USER01 to link in READ mode to any of USER02's minidisks:

```
TSS PERMIT(USER01) VMMDISK(USER02)
```

This example revokes READ access to USER02's minidisk:

```
TSS REVOKE(USER01) VMMDISK(USER02)
```

VMNODE Resource Class—Secure z/VM Nodes

Valid on z/VM.

Use VMNODE to secure z/VM nodes (RSCS Network Node prefixes).

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) VMNODE(`*`oper`*`,...)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) VMNODE(`*`oper`*`,...)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The VMNODE ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The VMNODE ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- VMNODE(OWN) authority to ADD or REMOVE nodes from ACIDs
- VMNODE(XAUTH) authority to permit or revoke access to VMNODEs

The administrator can use any of the following methods to control access to VMNODEs: Expiration, Facility, Time/Day, and Actions.

Examples: VMNODE resource class

This example protects a VMNODE called JACKSON, by assigning ownership to the Corporate Level Division:

```
TSS ADDTO(CORPDIV) VMNODE(JACKSON)
```

This example removes ownership of VMNODEs:

```
TSS REMOVE(CORPDIV) VMNODE(JACKSON)
```

This example permits a user in the Accounts Payable Department to transfer data to RSCS nodes in Dallas and Miami:

```
TSS PERMIT(APUSER) VMNODE(DALLAS,MIAMI)
```

This example revokes access to the RCSC node in Miami:

```
TSS REVOKE(APUSER) VMNODE(MIAMI)
```

VMRDR Resource Class—Secure Virtual Machine Readers

Valid on z/VM.

Use VMRDR to secure virtual machine readers.

Note: z/VM Readers control the spooling or transferring of virtual spool files to specific virtual machines, from a specified ACID.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) VMRDR(oper,...)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

Generic prefixing allows the administrator to group a set of similar resource names together, and define them by generic prefix.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) VMRDR(oper,...)
```

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The VMRDR ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The VMRDR ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- VMRDR(OWN) authority to ADD or REMOVE VMRDRs from ACIDs
- VMRDR(XAUTH) authority to permit or revoke access to VMRDRs

The administrator can use any of the following methods to control access to VMRDRS: Expiration, Facility, Time/Day, and Actions.

Examples: VMRDR resource class

This example protects a CMSBATCH machine by assigning ownership of its reader to the Development Department:

```
TSS ADDTO(DEVDEPT) VMRDR(CMSBATCH)
```

This example removes ownership:

```
TSS REMOVE(DEVDEPT) VMRDR(CMSBATCH)
```

This example permits a group of users in the Development Department to submit jobs to the CMS batch processor:

```
TSS PERMIT(PROFDEV) VMRDR(CMSBATCH)
```

This example revokes access to CMSBATCH:

```
TSS REVOKE(PROFDEV) VMRDR(CMSBATCH)
```

VOLUME Resource Class—Secure DASD and Tape

Valid on z/OS, z/VSE, and z/VM.

Use VOLUME to secure DASD or tape volumes.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) VOLUME(entries)
```

Length of entries

Two to six characters. Entries are treated as prefixes only if the generic indicator (G) follows each entry.

Capacity of list

One to 30 entries per TSS command

Generic prefixing allows the administrator to group a set of similar resource names together, and define them by generic prefix.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) VOLUME(oper,oper,...)
                 ACCESS(access levels)
```

Length of entries

Two to six characters. Entries are treated as prefixes only if the generic indicator (i.e.,(G)) is suffixed to the entry.

Capacity of list

One to 30 entries per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The VOLUME ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The VOLUME ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- VOLUME(OWN) authority to ADD or REMOVE volume ownership from ACIDs
- VOLUME(XAUTH) authority to permit or revoke access to VOLUMEs

The administrator can:

- Specify the access levels: ALL, BLP, CONTROL, CREATE, NOCREATE, NONE, READ, SCRATCH, and UPDATE. If access in not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to VOLUMEs: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Vol/Ser Attributes

The volume serial number or prefix must be followed by a (G) to indicate a generic volume prefix.

Note: The use of attributes is an administrative way to document that the volume relates to a disk or tape. CA Top Secret will honor the rule for both disk and tape access requests if they occur.

Examples: VOLUME resource class

This example protects all tape volumes with the generic prefix of 10000, by assigning ownership to the Corporate Department:

```
TSS ADDTO(CORP4) VOLUME(10000(G))
```

The administrator may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORP4) VOLUME(10000(G))
```

This example permits all users to access storage volumes:

```
TSS PERMIT(ALL) VOLUME(STOR01,STOR02,STOR03) ACCESS(CREATE)
```

This example revokes access to storage volumes:

```
TSS REVOKE(ALL) VOLUME(STOR01,STOR02,STOR03)
```

VSELIB Resource Class—Secure TSSVSE Libraries

Valid on z/VSE.

Use VSELIB to secure VSELIB for TSSVSE libraries

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) VSELIB(VSELIBR1)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) VSELIB(VSELIBR1.ABCDEFGH)`

Prefix length

One to forty-four characters

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The VSELIB ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The VSELIB ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- VSELIB (OWN) authority to ADD or REMOVE ownership of VSELIB resources from ACIDs
- VSELIB(XAUTH) authority to permit or revoke access to VSELIB resources

The administrator can use any of the following methods to control access to VSELIB resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Example: VSELIB resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) VSELIB(PRODLIBR1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) VSELIB(PRODLIBR1)
```

This example permits users in the Technical Services Department to access VSELIBR1.ABCDEFGH on Fridays only:

```
TSS PERMIT(TECHPROF) VSELIB(VSELIBR1.ABCDEFGH))
                     DAYS(FRIDAY)
```

This example permit users in the Technical Services Department to access VSELIBR1.ABCDEFGH:

```
TSS PERMIT(TECHPROF) VSELIB(VSELIBR1.ABCDEFGH)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) VSELIB(VSELIBR1.ABCDEFGH)
```

VSEPART Resource Class—Secure Partitions

Valid on z/OS and z/VSE.

Use VSEPART to secure VSEPART for z/VSE partitions.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*acid*`) VSEPART(VSEPART1)`

Prefix length

One to two characters

Capacity of list

One to two prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*acid*`) VSEPART(VSEPART1)`

Prefix length

One to two characters

MAXOWN length

2

MAXPERMIT length

2

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The VSEPART ACID types User, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The VSEPART ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- VSEPART (OWN) authority to ADD or REMOVE ownership of VSEPART resources from ACIDs
- VSEPART(XAUTH) authority to permit or revoke access to VSEPART resources

The administrator can use any of the following methods to control access to VSEPART resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Example: VSEPART resource class

This example protects the resource by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) VSEPART(vsepart)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) VSEPART(vsepart)
```

This example permits users to access VSEPART(BG) on Fridays only:

```
TSS PERMIT(TECHUSER) VSEPART(BG) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access VSEPARTR1.ABCDEFGH

```
TSS PERMIT(TECHPROF) VSEPART(BG)
```

This example revokes access:

```
TSS REVOKE(TECHUSER) VSEPART(BG)
```

VSEUSER Resource Class—Secure TSSVSE Users

Valid on z/VSE.

Use VSEUSER to secure VSEUSER for TSSVSE users.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) VSEUSER(PRODUSER)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) VSEUSER(PRODUSER)`

Prefix length

One to eight characters

Capacity of list

One to eight prefixes per TSS command.

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The VSEUSER ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The VSEUSER ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- VSEUSER (OWN) authority to ADD or REMOVE ownership of VSEUSER resources from ACIDs
- VSEUSER(XAUTH) authority to permit or revoke access to VSEUSER resources

The administrator can use any of the following methods to control access to VSEUSER resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: VSEUSER resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) VSEUSER(PRODUSER)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) VSEUSER(PRODUSER)
```

This example permits users in the Technical Services Department to access IMS1 on Fridays only:

```
TSS PERMIT(TECHPROF) VSEUSER(PRODUSER) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access PRODUSER.

```
TSS PERMIT(TECHPROF) VSEUSER(PRODUSER)
```

This example revokes access:

```
TSS REVOKE(TECHPROF) VSEUSER(PRODUSER)
```

VTAMAPPL Resource Class—Secure VTAM ACB Access

Valid on z/OS and z/VSE.VTAM ACB

Use VTAMAPPL to secure access to VTAM ACB's under a non-APF authorized program.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) VTAMAPPL(`*`acb-name`*`,`*`acb-name`*`...)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) VTAMAPPL(`*`acb-name`*`,`*`acb-name`*`)`

Prefix length

One to eight characters

Capacity of list

One to five prefixes

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The VTAMAPPL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The VTAMAPPL ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- VTAMAPPL(OWN) authority to ADD or REMOVE non-APF-authorized access of VTAM ACBs from ACIDs
- VTAMAPPL(XAUTH) authority to permit or revoke non-APF-authorized access of VTAM ACBs

The administrator can:

- Specify the access levels NONE and READ.
- Use any of the following methods to control access to VTAMAPPL resources: Expiration, Facility, Time/Day, and Actions.

Examples: VTAMAPPL resource class

This example gives NETPGMR ownership of ACBs defined for some test VTAM applications that have been defined as TSTAPPL and NEWACB:

```
TSS ADDTO(NETPGMR) VTAMAPPL(TSTAPPL,NEWACB)
```

This example removes ownership:

```
TSS REMOVE(NETPGMR) VTAMAPPL(TSTAPPL,NEWACB)
```

This example permits ACID NETLINK which is assigned to a batch job that executes a non-APF-authorized program that opens the above listed VTAM ACB names:

```
TSS PERMIT(NETLINK) VTAMAPPL(NEWACB,TSTAPPL)
```

```
TSS PERMIT(ALL) VTAMAPPL(NEWACB,TSTAPPL)
                ACCESS(NONE)
```

WRITER Resource Class—Secure Job Output

Valid on z/OS.

Use WRITER to secure the monitoring, routing, and processing of job output to local devices, RJE stations or NJE nodes.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) WRITER(`*`oper`*`,`*`oper`*`...)`

Prefix length

Two to eight characters

Capacity of list

One to eight prefixes per TSS command

When used with TSS PERMIT/REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) WRITER(`*`prefix(es)`*`)`

Prefix length

Two to eight characters

Capacity of list

One to eight prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOHAS, and WHOOWNS
- The WRITER ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE
- The WRITER ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE
- WRITER(OWN) authority to ADD or REMOVE ownership of WRITER from ACIDs
- WRITER(XAUTH) authority to permit or revoke access of the WRITER resource

The administrator can:

- Specify the access levels: ALL, NONE, READ, CONTROL, and UPDATE. If ACCESS is not specified, CA Top Secret defaults to READ access.
- Use any of the following methods to control access to WRITER: Expiration, Facility, Time/Day, and Actions.

Note: Only z/OS, ESA release 3.1.3 and above offer support for all resources.

Masking

The WRITER resource class supports all masking characters.

Examples: WRITER resource class

This example assigns ownership to an ACID, USER01, so that it can have routing control to all printers at a specific node:

```
TSS ADDTO(USER01) WRITER(NODE01.LOCAL.PRT6)
```

This example removes ownership:

```
TSS REMOVE(USER01) WRITER(NODE01.LOCAL.PRT6)
```

This example permits USER01 access to control output printing for printer 7:

```
TSS PERMIT(USER01) WRITER(NODE01.LOCAL.PRT7)
```

This example revokes access:

```
TSS REVOKE(USER01) WRITER(NODE01.LOCAL.PRT7)
```

XCSFKEY Resource Class—Secure Exportation of ICSF Cryptographic Keys

Valid on z/OS.

Use XCSFKEY to secure the exportation of z/OS Integrated Cryptographic Service Facility (ICSF) cryptographic keys. For information on setting the keys, see IBM's *OS/390 Integrated Cryptographic Service Facility: Administration Guide*.

When used with TSS ADDTO/REMOVE, this resource class has the following format:

```
TSS ADDTO(acid) XCSFKEY(XCSFKEY1)
```

Prefix length

1–26 characters

Capacity of list

1–5 prefixes per TSS command.

When used with TSS PERMIT/REVOKE, this resource class has the following format:

```
TSS PERMIT(acid) XCSFKEY(XCSFKEY1)
```

Prefix length

1–246 characters.

This resource class is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS.
- The ACID types User, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADD/REMOVE.
- The ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT/REVOKE.
- CSFKEYS (OWN) authority to ADD or REMOVE ownership of CSFKEYS resources from ACIDs.
- CSFKEYS(XAUTH) authority to PERMIT or REVOKE access to CSFKEYS resources.

The administrator can use any of the following methods to control access to CSFKEYS resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: XCSFKEY resource class

This example protects the resource with CA Top Secret by assigning ownership to the Corporate Department(owning acid):

```
TSS ADDTO(CORPORAT) XCSFKEY(XCSFKEY1)
```

The administration may now PERMIT access to users or profiles that require access.

This example removes ownership:

```
TSS REMOVE(CORPORAT) XCSFKEY(XCSFKEY1)
```

This example permits users to access XCSFKEY(XCSFKEY1) on Fridays only:

```
TSS PERMIT(TECHUSER) XCSFKEY(XCSFKEY1) DAYS(FRIDAY)
```

This example permits users in the Technical Services Department to access XCSFKEY1:

```
TSS PERMIT(TECHPROF) XCSFKEY(XCSFKEY1)
```

This example revokes access :

```
TSS REVOKE(TECHUSER) XCSFKEY(XCSFKEY1)
```

ZMFAPLA Resource Class—Secure z/OSMF Authorization Roles

Valid on z/OS.

Use ZMFAPLA to secure access to z/OSMF links and tasks.

When used with TSS ADDTO or REMOVE, this resource class has the following format:

`TSS ADDTO(`*`acid`*`) ZMFAPLA(`*`role`*`)`

Prefix length

2-26 characters

Capacity of list

1-8 prefixes per TSS command

When used with TSS PERMIT or REVOKE, this resource class has the following format:

`TSS PERMIT(`*`acid`*`) ZMFAPLA(`*`role`*`)`

Prefix length

1-246 characters

Capacity of list

1-8 prefixes per TSS command

This keyword is used with:

- The commands CREATE, ADDTO, REMOVE, PERMIT, REVOKE, ADMIN, DEADMIN, WHOOWNS, and WHOHAS
- The ZMFAPLA ACID types User, Profile, Department, Division, Zone, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS ADDTO or REMOVE
- The ZMFAPLA ACID types User, Profile, DCA, VCA, ZCA, LSCA, SCA, and MSCA when used with TSS PERMIT or REVOKE
- ZMFAPLA(OWN) authority to ADD or REMOVE ownership of ZMFAPLA resources from ACIDs
- ZMFAPLA(XAUTH) authority to PERMIT or REVOKE access to ZMFAPLA resources

An administrator can use any of the following methods to control access to ZMFAPLA resources: Expiration, Facility, Program Pathing, Time/Day, and Actions.

Examples: ZMFAPLA resource class

This example protects the resource by assigning ownership to the Corporate Department. After ownership is assigned, the administrator can PERMIT access to users or profiles.

```
TSS ADDTO(CORPORAT) ZMFAPLA(ZOSMF)
```

This example PERMITs a user to access the DASD management class in z/OSMF:

```
TSS PERMIT(JOHN01) ZMFAPLA(ZOSMF.DASD_MANAGEMENT.DASD_MANAGEMENT)
```

This example revokes access to DASD management:

```
TSS REVOKE(JOHN01) ZMFAPLA(ZOSMF.DASD_MANAGEMENT.DASD_MANAGEMENT)
```